DESIGN AND PRODUCTION OF A 3-D PRINTED WIRELESS HEXAPOD

A Major Qualifying Project Report:

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Bachelor of Science

by

Andrew Beaupre

Tyler Collins

Aras Nehir Keskin

Cody Woodard-Wallace

Date: April 30th 2015

Approved By:

____

Prof. David C. Planchard, Project Advisor

Table of Contents

Table of Contents i

Table of Figures iii

Table of Tables v

Nomenclature vi

Abstract vii

Introduction 1

Background 2

Components of a Hexapod 2

Gait Styles 3

Benchmarking 4

RHex Robot 6

DASH Robot 8

RoACH Hexapod Robot 10

Methods and Procedure 13

Design Goals and Project Objectives 13

Design Specifications 13

Gait Style Selection 14

Conceptual Designs 16

Design 1 16

Design 2 17

Selection of Final Design 18

Electrical Design 19

Arduino Controller 19

Motors and Motor Drivers 21

Communication 23

Power 27

Range Testing 29

Leg Orientation Issues 31

Materials Research 32

Assessing the Viability of 3-D Printing Materials 32

Heat Testing 33

Rapid Prototyping 35

Introduction 35
Printed Parts Overview 36
Additive Manufacturing of the Parts 37
Frame (Part no. 1) 37
Universal Joint Rings (Part no. 2) 40
Ball-Socket Joint (Part no. 3) 42
Lower Leg (Part no. 4) 43
Idler Gear and Spur Gear (Part no. 5 and 6) 44
Gear Mount (Part no. 7) 45
Driving Gear (Part no. 8) 46
Motor Mount (Part no. 9) 48
Conclusion 50
Controller Box 50
Calculations 52
Expected mass of 3-D printed components 52
Forces due to the weight of the body and additional components 52
Force acting on each leg due to weight of robot 53
Finite Element Analysis – Stress Concentrations 54
Reaction forces due to weight of the robot 56
Frictional Force (Static) 57
Frictional Force to Overcome (Dynamic) 57
Torque Required to Overcome the Force from Weight 58
Factor of Safety of the Motor 59
Motor Selection 59
Conclusions 61
Prototype Assessment 61
Future Work 62
Appendices 64
Appendix A: Program Code 64
Hexapod Code 64
Controller Code 67
Appendix B: Heat Testing 69
Appendix C: Hexapod Model SolidWorks Parts 77
Appendix D: Gear Motor Comparison 78

Table of Figures

Figure 1: Robotic hexapod with 3 DOF per leg 1

Figure 2: Basic components of a hexapod robot 3

Figure 3: Rhex Robot 6

Figure 4: Dash robot 8

Figure 5: RoACH hexapod (quarter for scale) 10

Figure 6: Diagram of RoACH's tripedal gait style at four different maximum positions 11

Figure 7: Grouping of legs in cyclic tripedal gait 15

Figure 8: Movement phases of legs 15

Figure 9: CAD model of design 1 16

Figure 10: CAD model of design 2 (not all components shown) 17

Figure 11: Detailed view of joints used in design #2 18

Figure 12: Rendered model of final design 19

Figure 13: Flow diagram of Arduino program 20

Figure 14: Two channel encoder output 22

Figure 15: Single channel of encoder at 75 RPM 23

Figure 16: Wireless standards and features 25

Figure 17: Mesh networking example 25

Figure 18: Point-to-multipoint networking example 26

Figure 19: Hexapod battery 29

Figure 20: Example X-CTU range test results with 0.5 m separation 30

Figure 21: X-CTU range test graphs with modules outside 30

Figure 22: Comparison of (A) leg start location, (B) position after one rotation, and (C) position after two rotations 31

Figure 23: Typical material & application properties of Ingeo Biopolymer 4043D 32

Figure 24: Entire labeled model. Arrows on the motors show where the shaft enters the gear mount and spur gear. 37

Figure 25: Frame split, printing method #1 38

Figure 26: Warping occurring during printing process 38

Figure 27: Whole frame before division (left), Frame divided into 4 segments to prevent warping (right) 39

Figure 28: Mating of the frame components 40

Figure 29: Universal joint to frame connection 41

Figure 30: Universal joint ring tolerancing 41

Figure 31: CAD model of the 3-D printed ball and socket joint 43

Figure 32: Connection achieved using adhesives, by assembling in the position of the red arrow. The ball-socket joint is on the left, connecting to the lower leg on the right. 44

Figure 33: Idler (left) and spur (right) gear print orientation 45

Figure 34: Gear Mount CAD models. (A): inside surface facing the interior of robot. (B): opposite surface facing outwards. 46

Figure 35: Driving gear with highlighted D-profile hole for motor shaft entry 46

Figure 36: Left: technical drawing of the DC motor – dimensions in mm, motor shaft is 8 mm. Right: diagram of motor connecting to the gear mount, which then connects to the driving gear on the other side. The letter X indicates the shortage of the shaft. 47

Figure 37: Motor to gear mount - driving gear top view. The motor shaft is in the extruded cylinder, which goes through the largest hole on the gear mount. 48

Figure 38: Motor mount (red) attached to the motors using Gorilla Tape (CAD representation) 49

Figure 39: Bottom view of the frame. The red motor mount is connected to the frame from four places using nut-bolt fastening. 49

Figure 40: Exploded view of wireless controller (CAD) 50

Figure 41: Leg diameter 54

Figure 42: Von Mises stress simulation 55

Figure 43: Inner and outer leg dimensions 56

Figure 44: Rotating Gear Radius 58

Figure 45: Side view of finished prototype 61

Figure 46: Front view of finished prototype 61

Figure 47: Diagram of multiple gait styles 62

Table of Tables

Table 1: Assessment criteria for benchmarking 5

Table 2: Standardized assessment of RHex robot 7

Table 3: Standardized assessment of DASH robot 9

Table 4: Standardized assessment of the RoACH Robot 12

Table 5: Design specification for Hexapod 13

Table 6: Motor specifications 21

Table 7: Motor driver specifications 21

Table 8: XBee 802.15.4 module information 26

Table 9: Power draw of hexapod components 28

Table 10: Power calculations to find battery capacity 28

Table 11: Specific tensile strength of various materials 33

Table 12: 3-D printed parts overview. (*): 1 part printed in 4 segments 36

Table 13: Expected Mass of Prototype Components 53

Nomenclature

m (Mass)

kg (Kilograms)

g (Gravitational acceleration)

m (Meters)

A (Area)

yds. (Yards)

r (Radius)

Pa (Pascals)

μ (Coefficient of friction)

V (Volts)

Rpm (Revolutions per minute)

h (Hours)

L x W x H (Length, Width, Height)

g (Grams)

V (Volume)

(‘) (Feet)

σ (Stress)

oz. (Ounces)

π (Pi: ~3.14159)

N (Newtons)

N (Normal force)

mA (Milli amperes)

F_r (Frictional force)

dB (Decibel)

dBm (Decibel-milliwatts)

ρ (Density)

(“) (Inches)

W (Weight)

mm (Millimeters)

F (Force)

s (Seconds)

d (Distance)

ω (Angular velocity)

τ (Torque)

Hz (Hertz)

t (Time)

mAhrs (Milli amp-hrs)

Abstract

The purpose of this MQP was to research, analyze, and construct a working prototype of a low-cost, reliable, serviceable, modular, wirelessly-controlled, terrestrial hexapod (six-legged robot) for research usage. The dynamic goal of the prototype was to achieve the capability of forward, backward, and turning motions while using low cost additive manufacturing methods to produce all of the mechanical components for open source technology. The prototype was completed and tested by an interdisciplinary team of electrical and mechanical engineering students and required collaborative design across two engineering disciplines. 3-D printing was used as the central additive manufacturing method, including technologies such as Fused Deposition Modelling (FDM) and Photopolymerization. Printed parts ranged in complexity from simple gears to intricate joint assemblies. The key focus of the project was designing for manufacturability in real-time applications.

Introduction

In the rapidly advancing field of modern robotics, researchers continue to look for design inspiration in nature. The field, dubbed biomimetics or biomimicry, strives to emulate biological processes and elements from nature to help solve human problems (Benyus, 1997). By observing these systems and emulating them, scientists and researchers can produce new products in a multitude of fields such as medicine, materials science, nanotechnology, and industrial design. More recently, researchers have used biomimicry to create biologically inspired styles of robotic locomotion which produce motion in higher degrees of freedom (DOF) in an effort to satisfy increasingly complex movement requirements. These motion types can include anything from walking or jogging to more complicated movements such as climbing, jumping, or limbless locomotion. Prominent examples of biologically inspired locomotion which are being researched today are hexapods (six legged robots, see Figure 1) and octopods (eight-legged robots), which typically range from one to three DOF per leg (depending on their complexity) and can be used to navigate over uneven terrain or obstacles on the ground.

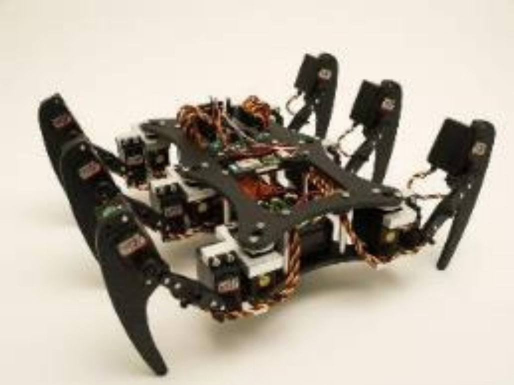


Figure 1: Robotic hexapod with 3 DOF per leg

In an effort to create an inexpensive, open source, wireless hexapod, the team aimed to 3-D print all custom mechanical components required for prototype production, significantly reducing the overall cost of the project. A secondary goal of the project was to design a mechanical motion system which could produce multiple gait styles without requiring additional components. After conducting background research through benchmarking, the team created preliminary designs in the SolidWorks Computer Aided Design (CAD) program. Each of these designs were assessed based on their ability to satisfy project objectives (see Design Goals and Objectives), and a functional prototype was manufactured using the specifications of the final design.

Background

Components of a Hexapod

A hexapod consists of a number of subsystems that must work together to allow the robot to achieve its desired functionality. A hexapod must have six legs, a drive system to allow movement of the legs, an on-board controller, and possibly a variety of sensors to monitor, observe, and analyze the robot's environment in real time.

The complexity of the drive system is directly related to the environment that the robot will function in and its design tasks or objectives. Many hexapods that can be found on the market rely on a number of servo motors as the primary actuation method for each leg of the hexapod. These servo motors serve as the mechanical joints of each leg. Figure 2 shows a hexapod with a servo controlling the forward and backward movement of the entire leg and an additional servo at one of the joints in the leg which, when actuated, extends the reach of the leg. The added range of motion and DOF of each leg of the robot adds to the complexity of the mechanical drive system and the method of controlling the drive system from a software standpoint. Increasing the complexity of the locomotion system also increases the total cost of the robot. Because of this, cost restrictions can have an impact on the maneuverability of the robot and its ability to satisfy application-specific objectives.

The controller also varies greatly with the desired functionality of the hexapod. The hexapod shown in Figure 2 is driven by a controller specifically designed for driving multiple servo motors. Controllers used for autonomous applications require a larger capacity for computation since the robot must be able to read input from a variety of sensors, process that information into useable data, and then generate some form of response, all before the sensor inputs change. In an application where the robot will be remote controlled, the on-board processor will have less decision making to do, but will need to route information from the user to the appropriate subsystem and provide necessary feedback at the same time.

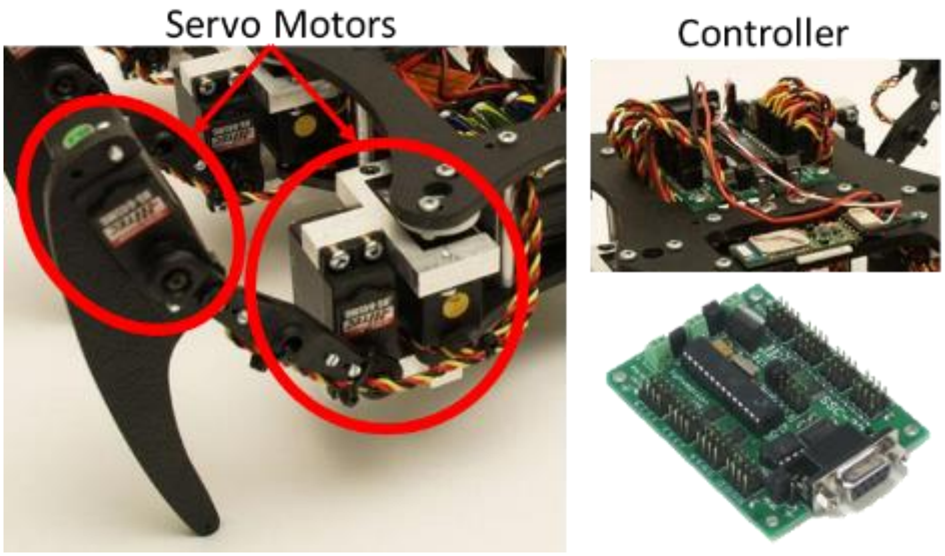


Figure 2: Basic components of a hexapod robot

Sensors provide a connection between the physical world and the processor on-board the robot or within the controller held by the user. Ultra-sonic sensors can be used to determine the distance between the front of a robot and an object in its path. Limit switches attached near the tip of the legs can show when a leg has encountered an obstacle.

Gait Styles

The hexapod walking motion, or gait, has a number of advantages over bipedal or wheeled locomotion styles. Although wheeled systems are typically faster on level ground, hexapedal locomotion produces the fastest movement of all legged robots. Unlike wheeled robots, hexapods can safely maneuver through uneven terrain and over smaller obstacles (Ding et al, 2010). Hexapods can also turn without requiring forwards or backwards motion, and maintain superior stability when walking by keeping their center of mass within a triangular area created by any three planted legs (Ding et al, 2010).

Hexapedal gait styles can be broken down into two subcategories; rectangular leg orientations and hexagonal leg orientations. In general, hexagonal leg orientations allow for more complex movements, easier turning, and increased stability when mobile. Alternatively, rectangular leg orientations can produce motion with simpler locomotion systems at the cost of sacrificing some stability. Because the goal of the project was to produce a hexapod which uses a simple locomotion style, the team chose to review only rectangular leg orientation gait styles.

The most common hexapedal gait style is the 3+3 “tripedal” gait, which separates the legs on either side of the robot into two triangular groups. Each group contains two outside legs on one side of the robot and one middle leg on the opposite side of the robot. During motion, the groups alternate moving forwards or backwards, ensuring that three legs are in contact with the ground at all times. This style of gait produces the fastest hexapedal motion (Ding et al, 2010).

A second type of hexapedal gait is the 4+2 “quadruped” gait. In this gait style, four legs are kept in contact with the ground at all times. The robot then cycles through moving two legs at a time. Similarly, hexapods can also use a 5+1 “one by one” gait style in which one leg is moved at a time. 4+2 “quadruped” and 5+1 “one by one” gait styles are more stable (Ding et al, 2010), but a disadvantage is that they travel at reduced speeds. As the number of legs being actuated together decreases, so too does the robot’s movement speed. This is because more cycles of motion are required to move all six legs in one full rotation (a full hexapedal stride).

Benchmarking

Before examining current products (hexapods) on the market, the team developed a list of attributes which were used to assess the strengths and weaknesses of each product. Analysis of each product using the standardized assessment list (which can be seen in Table 1) allowed for a simple, clear, and consistent comparison of product specifications. Additional information specific to each individual product but unrelated to the standard assessment criteria is also included where appropriate. In some instances, information on specific components (such as motor specifications or controller dimensions) could not be found because most of these robots are either being used for research purposes or are not commercially available.

Table 1: Assessment criteria for benchmarking

Specification	Description
Price	The price paid by the consumer for the product
Durability	How resistant the robot is to damage or wear
Mechanical Complexity	The number of degrees of freedom / motors that are required to move each leg
Control Mechanism Complexity	How difficult it is for the user to operate the robot
Physical Dimensions	The physical dimensions of the robot (length, width, height)
Weight	Weight of the robot
Material Composition	A breakdown of the components and materials used to build the robot
Speed	The range of speed at which the robot can travel
Battery Life	How long the robot can operate (at various load levels) before requiring a new/recharged battery
Operating Range	Maximum range in which the robot can be controlled wirelessly

RHex Robot

Pictured in Figure 3, RHex (short for Robotic Hexapod) is an autonomous sealed-body hexapod which operates in wet, muddy, and swampy conditions. RHex was designed and manufactured by Boston Dynamics, and despite not being available commercially, has been subjected to rigorous U.S. government-run testing. The robot, which was originally conceived through funding from the Defense Advanced Research Project Agency (DARPA) consortium, has also been adapted by a number of institutions including the Kodlab (a lab at the University of Pennsylvania School of Engineering and Applied Science).

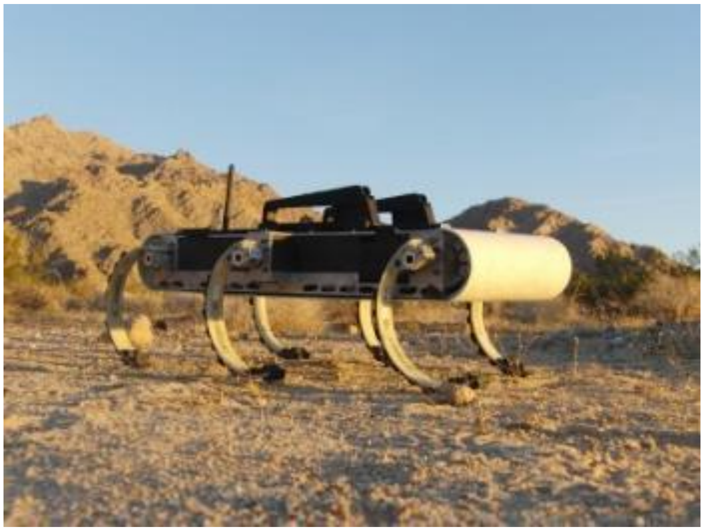


Figure 3: Rhex Robot

The RHex robot and its various adaptations are unique in that they have superior movement abilities when maneuvering over rough terrain and obstacles such as rocks, streams, snow, and steep inclines of up to 60 degrees. The current RHex design produced by Boston Dynamics includes two cameras which provide live video at 320x240 pixels while driving and 1280x960 pixels while still. Each of its six legs are independently controlled to allow for a wide variety of gait styles which are automatically selected by the onboard computer using feedback from each leg. Additionally, the robot is water submersible, and tolerant of salt, oil, and sand extremes. Table 2 summarizes the assessment specifications of the RHex robot.

Table 2: Standardized assessment of RHex robot

Rhex Benchmarking Assessment	
Price	N/A (not commercially available)
Durability	Excellent – the robot can successfully navigate through military combat landscapes such as forests and deserts without damage
Mechanical Complexity	Simple- Six motors (One per leg)
Control System	Difficult- Rhex is controlled through a series of sequences that are user inputted through a computer. Rhex is well known for its ability to simultaneously use all legs in preset motions, causing jumps.
Physical Dimension	56L x 41W x 13H cm (22” x 16” x 5.2”)
Weight	12.5 kg (27.5 lbs.)
Material Composition	Unavailable
Speed	0.9 m/s (3’/s)
Battery Life	6 hrs.
Operating Range	400 - 700 m (437 – 765 yds.) two way communication

DASH Robot

Pictured in Figure 4, DASH (short for Dynamically Autonomous Sprawled Hexapod) is an open body hexapod which operates in ideal weather conditions (since it has an open body, it cannot withstand any precipitation). DASH was designed and manufactured by University of California, Berkeley, and is available commercially. The robot was conceived based on inspiration from biomechanics, and uses an open loop tripedal style of locomotion.

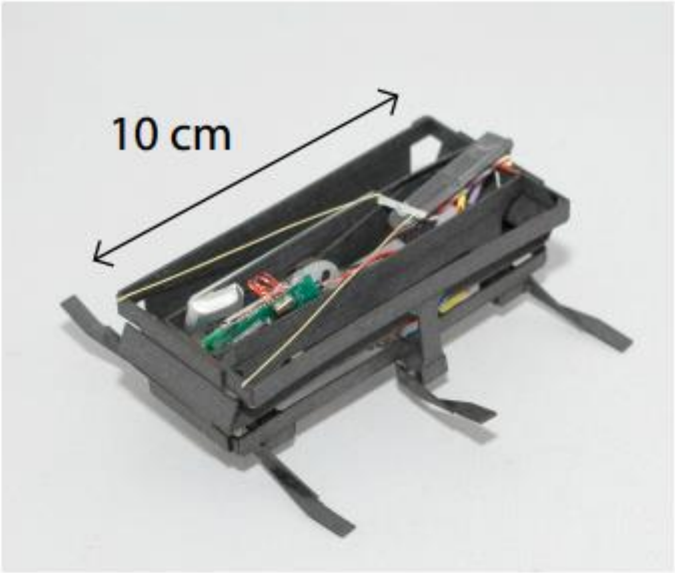


Figure 4: Dash robot

The DASH robot is unique in many ways, most notably for its manufacturing process. DASH is manufactured through use of SCM (Smart Composite Manufacturing), which makes use of a composited cardboard and fiberglass material that is laser cut and then folded into the correct shape. DASH's locomotion is described as "an open loop tri-gait system" which is controlled by a single motor and complex folds. The robot is capable of maneuvering by causing a small torque on one side which then causes that respective side to have more contact with the ground. This torque causes a U-turn that varies with friction (depending on the friction between the legs and the ground). Benchmarked specifications for the Dash Robot are summarized in Table 3.

Table 3: Standardized assessment of DASH robot

DASH Benchmarking Assessment	
Price	$49.99 (Including robot and controller)
Durability	Excellent - DASH was drop tested at a height of 15m (50 ft.) and drives away with minimal damage
Mechanical Complexity	Average - The motion of DASH is complex in terms of the folds and design, however it only uses one motor to drive all six legs
Control System	Simple - The control system has four basic movements: forwards, backwards, left and right
Physical Dimensions	10.6L x 5.08W x 5.08H cm (4”L x 2”W x 2”H)
Weight	16 g (0.035 lbs.) without batteries
Material Composition	Cardboard and Fiberglass
Speed	1.83 m/s (1’/s)
Battery Life	45 minutes (continuous use)
Operating Range	30 m (50 ft.) line of sight one-way communication

RoACH Hexapod Robot

The RoACH robot is one of the world’s smallest autonomous millirobots; its size is comparable to that of a quarter (see Figure 5). RoACH is manufactured using the SCM (smart composite microstructure) technique, which is a process that incorporates CO_2-laser micro machined composite fiber laminates with polymer films. The technique provides the structural integrity of rigid composite links and integrates compliant polymer hinges for mobility (Fearing et al., 2008). The greatest advantage of the SCM method is that it can provide a large number of joints for a very low cost; the robot has a total of 57 articulate joints. If the user wishes to add on joints, the cost of doing so is almost negligible due to the SCM process being formed from a single composite. The SCM manufacturing method utilizes elastic deformation of compliant hinges to provide joining. The joints can then be actuated with a small motor.

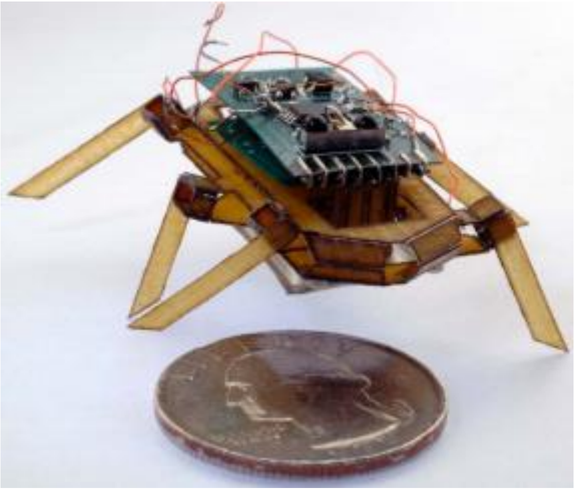


Figure 5: RoACH hexapod (quarter for scale)

RoACH utilizes a tripedal gait motion: three legs are coupled for synchronized motion (two outer legs on one side, one middle leg on the opposite side) for triangular stability using two linear actuators. The locomotion is supplied by the triangular pairing of the legs moving backwards, allowing the robot to leap forwards. Figure 6 illustrates the phases of motion used by the RoACH robot.

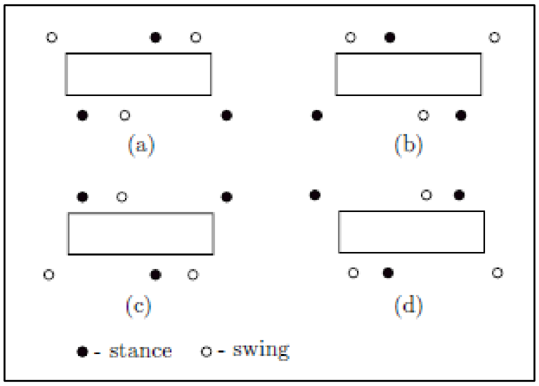


Figure 6: Diagram of RoACH's tripedal gait style at four different maximum positions

Actuation is achieved by the use of SMA (shape-memory alloy) wire (muscle wire). SMA actuation was chosen due to the compatibility with the SCM process, as well as being immensely lightweight and power dense. SMA actuation also provides single directional force which facilitates greater design freedom. RoACH is powered by a 20 mAh Lithium Polymer Battery which can provide just over nine minutes of run-time when the legs are continuously driven at 3 Hz. Additional specifications of RoACH are summarized in Table 4.

The wireless communication of the robot is obtained using bidirectional IrDA (infrared data association) infrared communication. This communication uses minimal battery power and the components required are very lightweight, however the operating range is limited to approximately one meter**.** Turning control is accomplished by the use of differential lengths for the legs. Left and right maneuver is controlled by timing of the actuation between the tripedal gait switches, which causes a moment that allows the robot to slip and turn.

Table 4: Standardized assessment of the RoACH Robot

RoACH Benchmarking Assessment	
Price	N/A (not commercially available)
Durability	Weak- RoACH is an extremely small robot, that when taking on any substantial force, bends or breaks
Mechanical Complexity	Average - The motion of RoACH is complex in terms of the folds and design, however it only uses two driving motors
Control System	Simple - The control system has four basic movements: forwards, backwards, left and right.
Maximum Physical Dimension	3 cm (1.2”)
Weight	2.4 g (0.85 oz.)
Material Composition	S2-Glass reinforced composites w/ flexible polymer hinges
Speed	1 body length per second (3 cm/s = 1.2”/s)
Battery Life	9 minutes (continuous use)
Operating Range	Standard: 1 m (3.3 ft.) line of sight, one way communication

Methods and Procedure

In this section, the methods and procedures of the mechanical design, electrical design, material selection, rapid prototyping, calculations, and communications and control of the hexapod are discussed.

Design Goals and Project Objectives

Key deliverables of the hexapod project were narrowed down to four project objectives. First, to design for simplicity by minimizing the number of actuators used to facilitate robot locomotion. Second, to use rapid prototyping to manufacture all major mechanical components. Third, to develop a wireless control system which will allow the robot to move forwards, backwards, and turn. And fourth, to use materials which are lightweight and durable, yet easily replaceable.

Design Specifications

Table 5 on this page and the following summarizes the design specifications of the hexapod prototype:

Table 5: Design specification for Hexapod

Design Specifications for HEXAPOD Robot			
Specification	Metric Unit	English Unit	Information
Weight	2270 g	5 lbs.	Measured with all components and accessories attached.
Length	0.216 m	8.5”	Measured from front most point (tip of frame to back most point (tip of frame).
Width	0.280 m	11”	Measured from left side leg tip to right side leg tip.
Height	0.229 m	9”	Measured vertically from leg tip to top most point on the antenna.

Max Speed	0.1524 m/sec	0.5 ft./sec	Specifications are based on movement on a flat plane.
Drive	N/a	N/a	2 Motor, 6 leg rotation
Communication	N/a	N/a	ADhoc Wifi network
Power Source (for what)	7.4V 5200 mAh	7.4V 5200 mAh	Electric
Material	N/a	N/a	Polylactide Resin (PLA)
Max. Operating Temperature	130°C (± 10°C)	266°F (± 18°F)	Specifications are based on motor limitations
Min. Operating Temperature	-20°C (± 10°C)	-4°F (± 18°F)	Specifications are based on motor limitations
Terrain	N/a	N/a	Flat ground (wood, cement, etc.)
Operating Distance	23.1 m	75.8 ft.	Specifications are based on Line of Sight
Price	$387	$387	Price includes all fasteners, electrical components, and printed components
Controller Box Size	13.3L x 8.9W x 5.1H cm	5.25"L x 3.5"W x 2"H	N/a
Controller Box Material	N/a	N/a	Acrylic
Controller Box Weight	0.23 kg	0.5 lbs.	Includes all electrical components

Gait Style Selection

After analyzing several different styles of hexapod locomotion through benchmarking research, the team chose to use a train-inspired style of locomotion for our hexapod. The implementation of a train-inspired locomotion style produced a modified tripedal gait in which the six limbs of our robot are broken down into two groups, each containing three legs. Figure 7 illustrates the separation of legs into their respective groups. The legs follow a circular motion similar to that of a row boat.

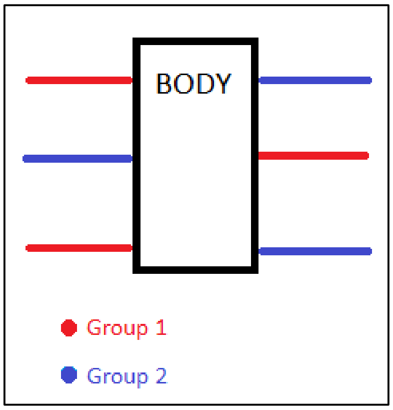


Figure 7: Grouping of legs in cyclic tripedal gait

The entire gait sequence of each leg can be broken down into four phases, depicted in Figure 8 and described below. Note that at any given time, the two groupings of legs are opposite each other in their respective phases. This means that when the three legs in group 1 are in the first phase of motion, the second three legs in group 2 are in the third phase of motion.

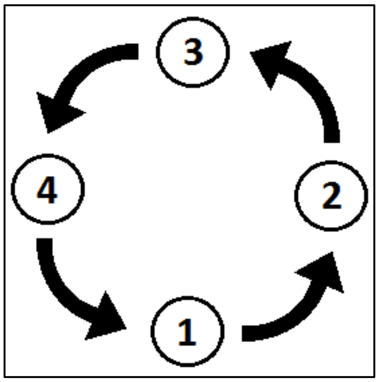


Figure 8: Movement phases of legs

- Phase 1: The leg is at its lowest position in the cycle and should be in contact with the ground.
- Phase 2: The leg is halfway into transitioning through the lifting process.
- Phase 3: The leg is now at its highest position relative to the ground.
- Phase 4: The leg is halfway into transitioning through the lowering process.

The decision to use this simplified locomotion style (as opposed to using one servomotor for each leg) brings with it a number of considerations which were addressed during materials selection and leg design. In most typical alternating tripedal gaits, the groups are controlled independently from each other. This allows for permanent contact on the ground as the first group of legs fully completes its four phases of motion before the second group begins its motion. With our modified cyclic tripedal gait design we sacrifice permanent ground contact for increased speed and simplicity. This puts a large amount of additional force on the leg joint, as there is a force of impact associated with the lurch of the robot in the direction it is traveling.

Conceptual Designs

Design 1

The first of the proposed designs (shown in Figure 9) incorporated a series of rigid inner chassis mounts which, when connected to a driving gear, would rotate around at fixed distances from the outer frame. The legs were also connected below the outside frame. Theoretically, the circular motion produced by the driving motors would translate centrally to the inner chassis and produce leg motion similar to that of the oar of a row boat, where the outside leg connection serves as a fulcrum facilitating a circular leg motion. The two driving motors (one for each side of the robot) were aligned back to back in the middle of the robot between the two chassis.

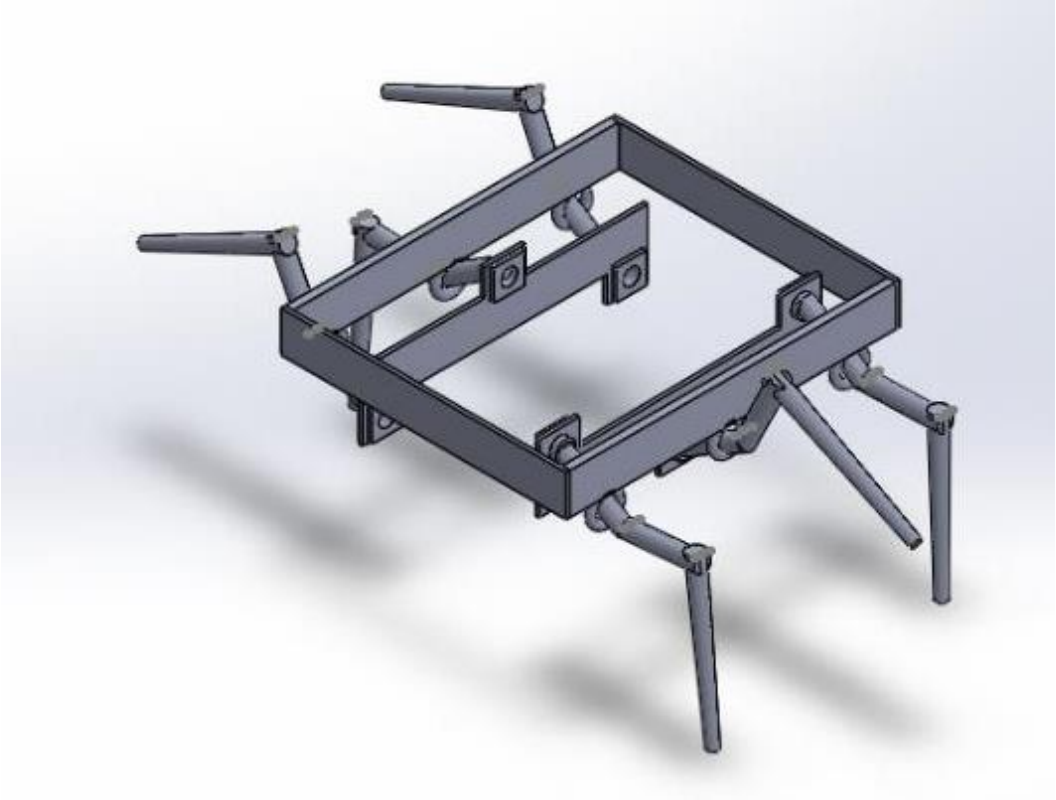


Figure 9: CAD model of design 1

Ultimately it was decided that the resulting leg motion of design 1 would not produce a viable gait style suitable for stable locomotion. Because of this, the design was discounted and other concepts were considered.

Design 2

The team's second design, shown in Figure 10, used a style of locomotion similar to, but slightly more complex than design 1. Once again, the locomotion produced was inspired by the motion of an oar while rowing. Rather than using a rotating chassis to produce the leg motion, design 2 utilized a gear chain assembly on each side of the robot to facilitate the motion of two sets of three driving gears. The gear and chain assemblies were attached to gear mounts which were fixed to the frame. By using a connecting chain, it was possible to rotate each gear assembly using a single motor. Legs of the robot were then connected to each of the driving gears with a ball and socket joint which allowed for unobstructed circular motion.

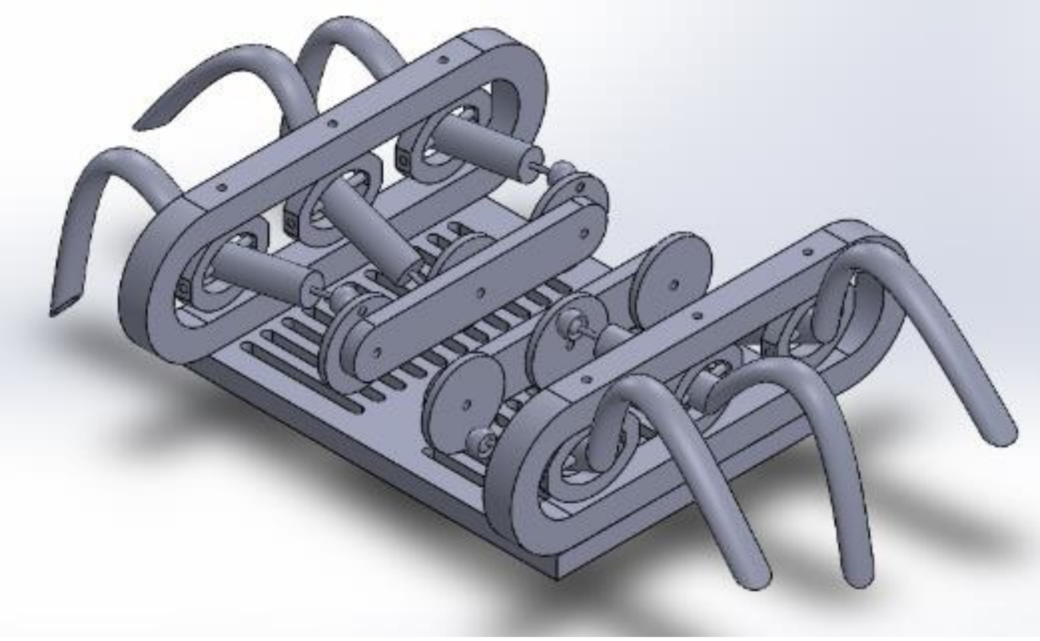


Figure 10: CAD model of design 2 (not all components shown)

The legs were then attached to the outer frame using a universal joint which acted as a fulcrum. Finally, due to the unaligned orientation of the inner gear mount in relation to the outer frame, the distance between the ball-socket and the universal joint varied as the driving gears rotated. The use of a cylindrical joint allowed for a change in length of the inner leg, eliminating the possibility of leg lock. Figure 11 provides a detail view of each of the three joints used in design 2.

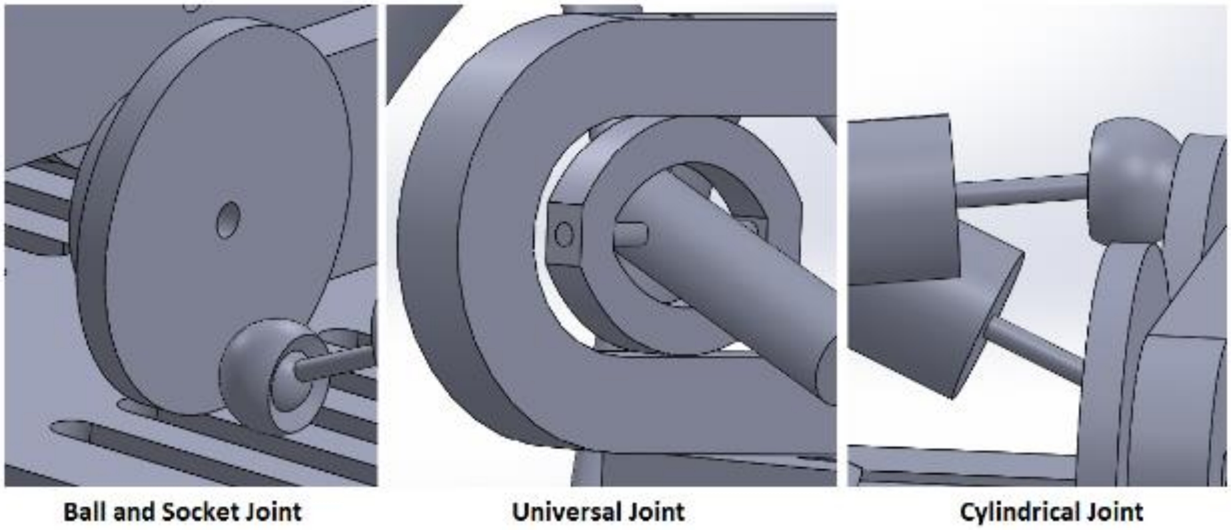


Figure 11: Detailed view of joints used in design #2

Design 2 was much more stable than design 1. The universal joint's orientation between the outer frame provided more protection than the leg-frame attachment point of design 1, which was located below the frame. When simulated in SolidWorks, design 2 successfully achieved the leg motion required for locomotion. The use of a universal joint produced a more dependable and protected fulcrum point of attachment to the outer frame of the robot than design 1. The switch from a rotating rigid chassis to a fixed gear mount better constrained the moving parts of the locomotion system to within the frame. Despite its superiority to design 1, design 2 incorporated a number of elements which made it complex to manufacture, increasing the cost of prototype production and the likelihood of system failure. These elements included the cylindrical joint used to vary the inner leg length, the gear and chain assembly used to drive the gears connected to the legs, and the ball-socket joint which connected the leg to the driving gears.

Selection of Final Design

After reviewing both conceptual designs, the team decided to alter the second design in an attempt to reduce its complexity and manufacturing cost. The result of this simplification was our final design, seen in Figure 12. Similar to the second conceptual design, the final design used the rotation of driving gears to produce the motion of the legs. Additionally, the team chose to keep the universal joints as the attachment method between the leg and the outer frame.

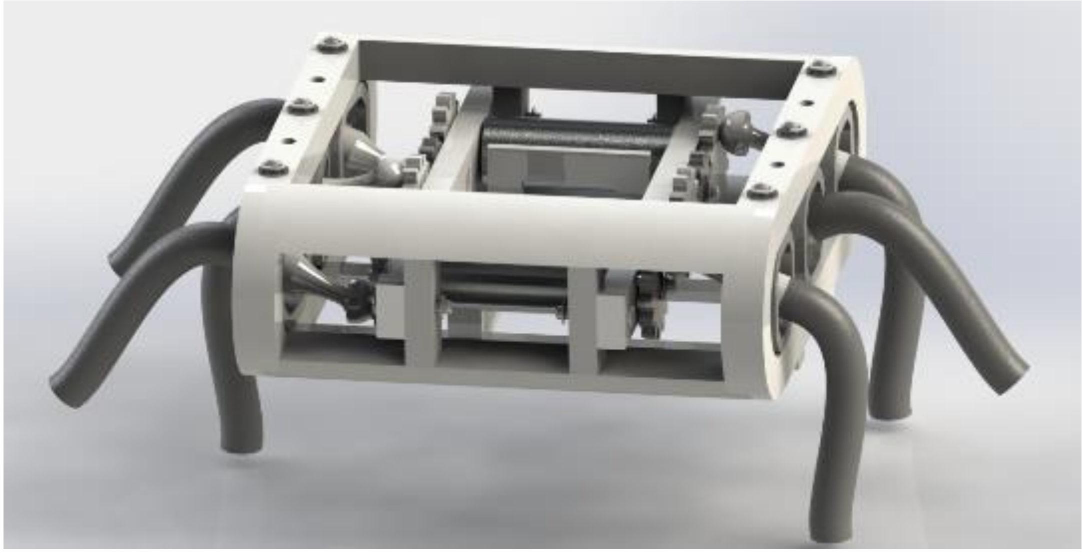


Figure 12: Rendered model of final design

One of the primary concerns with the second design was the need for a complex cylindrical joint in each leg. The team eliminated this requirement by changing the position of the gear mount in relation to the frame so that the central holes of the driving gears were aligned with the centers of the universal joints. This alignment ensured that the total length of the inner leg was consistent throughout the full revolution of the driving gear. The final design also replaced the gear chain of design 2 with a gear train, removing the chain and adding idler gears between the three driving gears to ensure synchronized motion. Unlike design 1, in which the motors had to be aligned centrally back to back, the use of a gear train allowed for the ability to stagger the motor locations on either side of the robot. By staggering the motors within the frame, the team was able to significantly reduce the frame width. To better utilize the inner space of the robot, the team chose to position the electronics between the two gear mounts.

Electrical Design

The electrical design of the hexapod was conducted in parallel with the design of the robot’s mechanical components. This was imperative for selecting the correct motors needed to actuate the legs of the robot. The electrical system can be broken down into four major components; the controller, the motors and motor drivers, the communication system, and the power supply.

Arduino Controller

The team chose to use an Arduino Uno based on the Atmel ATmega328 microcontroller to control the hexapod. This particular microcontroller is based on an 8-bit architecture and supports the 16 MHz oscillator on the Arduino board. The Arduino is programmed in its own development environment that supports C++ programming. Familiarity with the C programming language and the extensive support for Arduino based products and coding examples led to the decision to use an Arduino for the controller.

An Arduino board is used on both the controller, which is held by the user, and on the hexapod itself. The board on the hexapod is responsible for processing encoder output from the motors, feedback from the motor driving board, and incoming commands from the user. Figure 13 shows the flow of data from the user to the hexapod and how that data is handled by each Arduino board.

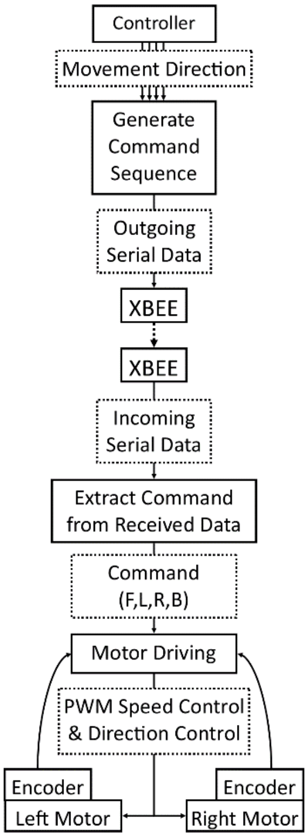


Figure 13: Flow diagram of Arduino program

The user presses buttons on the controller which generate a specific command sequence. Each of the command sequences is a string of three identical characters. Each movement command has a different character that is repeated. Forward is “F”, backward is “b”, left is “l”, and right is “R”. The command string is sent into the XBee for transmission as bytes of data so one character is transmitted at a time in rapid succession.

The characters are repeated within the command sequence as a basic method of error protection. Wireless transmission is not always perfect, so including the necessary data multiple times raises the possibility of it being received correctly by the receiver.

The XBee modules, which are discussed in more detail in the communication section, transmit the data to the hexapod’s XBee to be processed by that Arduino. The Arduino waits for data to come in, and upon arrival of the command sequence the characters are stored in a variable to be processed. It takes the entire command bytes and scans them for the characters “F”, “b”, “l”, or “R”. If one of these characters is found in that string, the Arduino turns the motors on in the correct directions. Once the motors have been turned on the Arduino counts the encoder pulses to turn them off once each side has independently reached a full rotation by using interrupt service routines that are triggered any time one of the encoders receives a rising edge. The code for both Arduinos can be found in Appendix A: Program Code.

Motors and Motor Drivers

The requirements of the motors were found to be a minimum torque of 52.5 oz·in and preferred rotational speed of 77RPM (see motor selection calculations section). Using these values a metal gear motor with a gear ratio of 75:1 was selected as the motor that would meet the requirements with some margin for error. Specifications of the motors and motor drivers are summarized in Table 6 and Table 7.

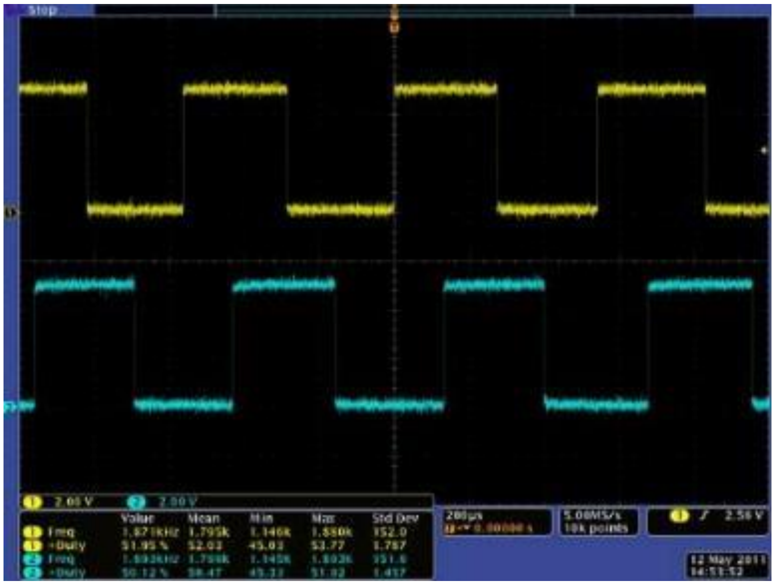
Table 6: Motor specifications

Motor: 75:1 Metal Gear-motor 25D (diameter) x 54L mm HP with 48 CPR Encoder	
Price	$36.95 each
Size	6.65L x 2.489W x 2.489H cm (2.62” x 0.98” x 0.98”)
Weight	102.625 g (3.62 oz.)
Operating Voltage	6 V
Operating Current	2 – 6 A

Table 7: Motor driver specifications

Motor Driver: Pololu Dual MC33926 Motor Driver Shield for Arduino	
Price	$29.95
Size	4.826L x 5.131W x 0.965H cm (1.90” x 2.02” x 0.38”)
Weight	20 g (0.705 oz.)
Operating Voltage	6-28 V
Operating Current	140 mA

The motor comes with the gearbox already attached to the output shaft. The total length of the motor fits between the two gear mounts. The selected model also has an encoder fixed to the end (opposite the output shaft) which uses a Hall Effect sensor to detect the rotation of a magnetic disk attached to the motor's shaft. The encoder has two channels, each detecting the rotation of the motor shaft, but are out of phase by 90° with respect to each other. This relationship can be seen in Figure 14 which is provided by Pololu as part of their documentation of the motor. The counts per revolution value (CPR) is determined by the square wave outputs of the encoder. Across the two channels there are 48 rising or falling edges to the square waves for each revolution of the motor's shaft.



	Value	Mean	Min	Max	Std Dev
1 Freq	1.871kHz	1.795k	1.146k	1.880k	152.0
1 +Duty	51.95 %	52.03	45.03	53.77	1.787
2 Freq	1.893kHz	1.799k	1.145k	1.802k	151.6
2 +Duty	50.12 %	50.47	45.22	51.02	1.457

Figure 14: Two channel encoder output

The actual CPR value differs from the given 48 due to the use of a gearbox. To find the actual encoder counts per single revolution of the output shaft the gear ratio must be multiplied by the motor's CPR:

$$motor\ CPR * gearbox\ ratio = actual\ counts\ per\ revolution$$

$$48\ CPR * 75 = 3600\ CPR$$

While the selected motor ships with two encoder channels, this application only required a single edge of a single channel. This reduced the effective counts per revolution from 48 (at the motor shaft) to 12 changing the total counts per revolution to 900. The oscilloscope image in Figure 15 shows a single channel of the encoder that is in use for our application as well as a measure of the frequency of transitions.

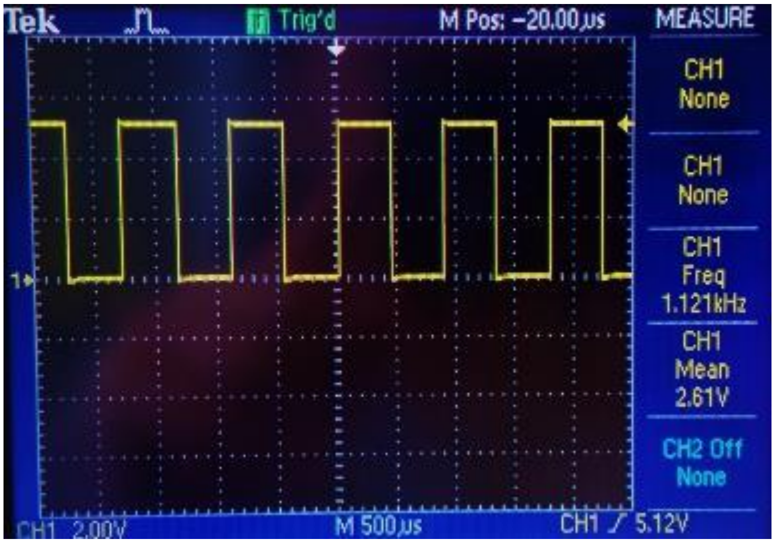


Figure 15: Single channel of encoder at 75 RPM

The frequency of the square wave measured in Figure 15 yields the speed that the motor output shaft is turning based on the following calculation:

$$encoder\ frequency * \frac{1\ revolution}{900\ counts} * \frac{60\ sec}{min} = revolutions\ per\ minute$$

$$1.121\ kHz * \frac{1\ revolution}{900\ counts} * \frac{60\ sec}{min} = 75\ RPM$$

The encoder outputs from each motor serve as inputs to the Arduino onboard the hexapod. Each rising edge of the square wave triggers an interrupt within the Arduino program which increments the count of that particular motor. Within the interrupt is a check for a value of 900 which corresponds to one full revolution of the output shaft, at which point the motor speed is set back to zero. The motors are turned on together, but are turned off independently when full rotation has been reached. This accounts for possible variances in speed of each motor and ensures the proper leg configuration will be maintained with each movement operation.

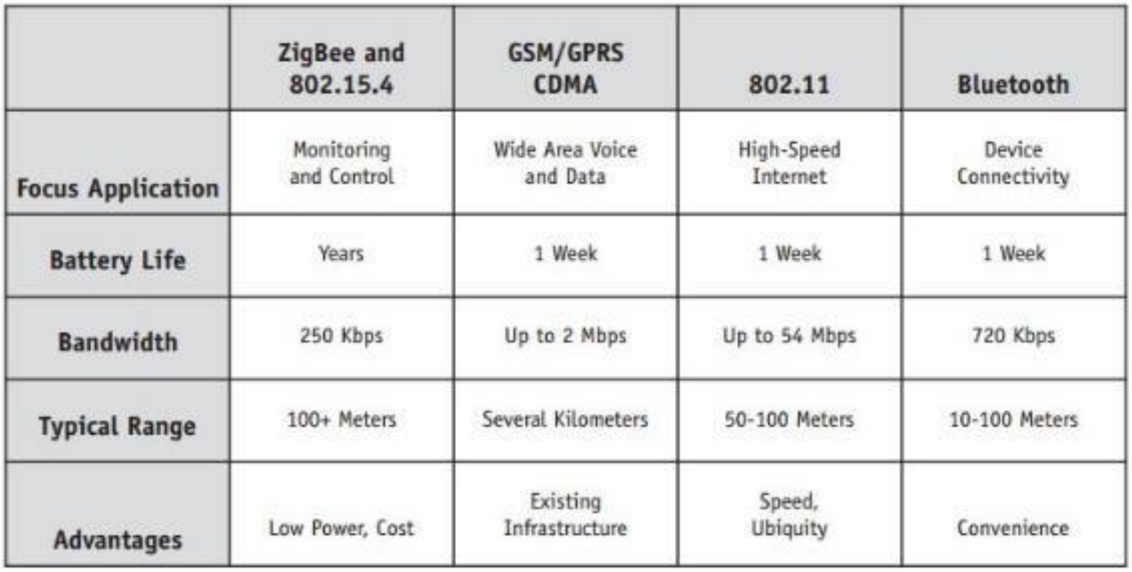
Communication

A reliable method of wireless communication between the user and the hexapod is essential for achieving the desired functionality of the robot. The desired specifications of this hexapod required an operating distance of 23.1 m (75.8 ft.), two way communication to facilitate movement controls to the robot, and interference protection from other external factors. From the desired specifications, a number of factors had to be kept in mind when selecting a method to fit the needs of this project.

There were many options on the market for wireless communication systems, each of which had different properties that made them good or bad for particular applications. The nature of our application brought forward a few properties that would be necessary to consider when making a selection including: low cost, low power consumption, long distance operation, reasonable data rates, and flexibility within a design.

A typical Wi-Fi network would meet most of these needs. Wi-Fi using the 802.11 IEEE standard can support data rates in the mega bit ranges (Mb), but requires access points covering the area of interest. Most other options required a more complex method of connecting the two devices together than would be necessary for a single point-to-point situation that we are using. For these reasons two XBee modules were used for this application.

XBee is a product of Digi International, a company specializing in wireless and radio frequency (RF) communication devices. XBee modules come in a variety of models which utilize different standards of communications including Wi-Fi, ZigBee, IEEE 802.15.4, and a proprietary mesh networking protocol. The Wi-Fi module is good for applications such as data logging or connecting devices together across an already existing network such as sensors in a factory. The 802.15.4 modules are good for point-to-multipoint or point-to-point networks, while the ZigBee modules can be used to create a mesh network for passing data over long distances by using many devices. Figure 16 compares ZigBee/802.15.4, cellular, Wi-Fi (802.11), and Bluetooth standards along with features and advantages to each. Figure 17 and Figure 18 show an illustration of a point-to-multipoint network and a mesh network example respectively (Digi, Demystifying 802.15.4 and ZigBee 2008).



	ZigBee and 802.15.4	GSM/GPRS CDMA	802.11	Bluetooth
Focus Application	Monitoring and Control	Wide Area Voice and Data	High-Speed Internet	Device Connectivity
Battery Life	Years	1 Week	1 Week	1 Week
Bandwidth	250 Kbps	Up to 2 Mbps	Up to 54 Mbps	720 Kbps
Typical Range	100+ Meters	Several Kilometers	50-100 Meters	10-100 Meters
Advantages	Low Power, Cost	Existing Infrastructure	Speed, Ubiquity	Convenience

Figure 16: Wireless standards and features

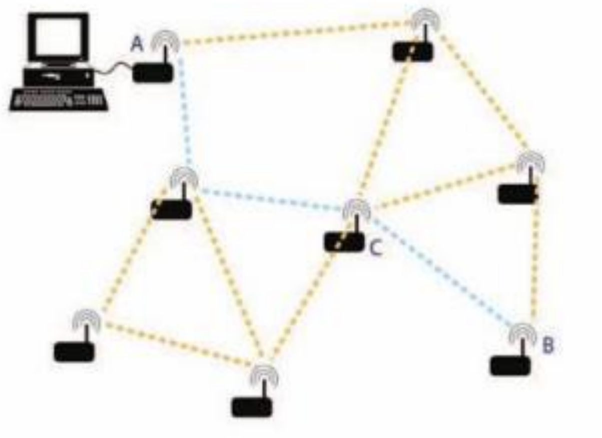


Figure 17: Mesh networking example

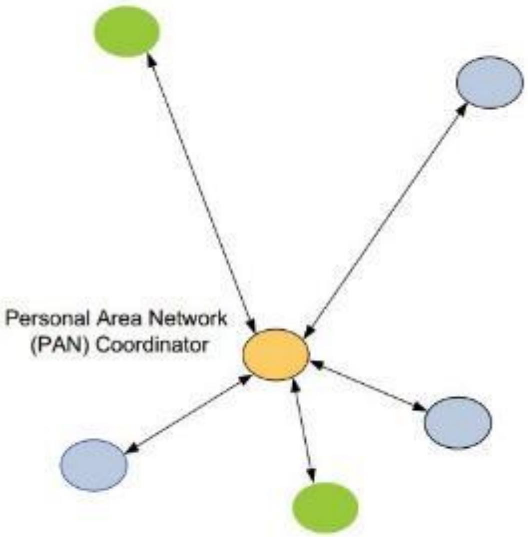


Figure 18: Point-to-multipoint networking example

The XBee module that best fit our needs was the one which conforms to the 802.15.4 standard without mesh networking. This particular module has the following key properties:

Table 8: XBee 802.15.4 module information

Range	~90 m
RF Data Rate	250 Kbps
Operating Frequency	2.4000-2.4835 GHz
Network Type	Point-to-Point
Wireless Standard	IEEE 802.15.4

The XBee communication operates on the Industrial Scientific and Medical (ISM) band which is unlicensed and can be used by the general public with a maximum transmission power of 1 W so long as the technology is using spread spectrum methods to prevent interference (FCC, 1996). Another benefit to using the XBee modules for our communication was the simplicity of their setup. Configuration of destination address for each module and the channel to be used were the only parameters that needed to be set for the two devices to be paired together for use. Configuration of the XBee modules was done through an XBee Explorer board which allows the user to interface to a computer and configure the XBee through the software program X-CTU (Digi, 2015).

The XBee module's user manual indicates that a received signal must have a strength of at least -92 dBm in order for the receiver module to be able to interpret the signal as something useable (XBee, 2015). The unit dBm is a power ratio in reference to 1 mW where the Decibel is a dimensionless unit used show a ratio between two measured powers. The maximum power that can be transmitted through the module is 1 mW or 0 dBm which provides a maximum power loss of 92 dB. To determine an estimate of the power loss that can be expected at our maximum distance of 30 m the following equation was used from Green and Obaidat (Green, Obaidat, 2002) which provides a propagation model for low height antennas:

$$P_{loss} = 40\,log(D) + 20\,log(F) - 20\,log(h_t h_t)$$

Where P_{loss} is the power lost in dB, D is the distance (in meters) between transmitter and receiver, F is the frequency of the transmitted signal in GHz, and $h_t h_t$ is the height of the transmitter and receiver in meters. 1.25 m is about the height a user would be holding the transmitter and 0.3 m is the maximum height the antenna would sit off the ground when attached to the hexapod.

$$P_{loss} = 40\,log(30\,m) + 20\,log(2.4\,GHz) - 20\,log((1.25\,m * 0.3\,m))$$

$$P_{loss} = 75.2\,dB$$

A power loss of 75.2 dB needs to be compared to the minimum received signal strength the XBee on the hexapod can accept to interpret the signal. The expected power loss of 75 dB can be subtracted from the transmission strength of 0 dBm to produce an expected received signal strength of -75 dBm which is greater than the minimum received power.

Power

Determining the power source for the hexapod first required knowledge of the power requirements of the components present in the design. Table 9 shows the typical power draw and the maximum power draw of the major components. The typical power draw refers to the expected amount of power that would be needed to operate that component under ideal or expected operating conditions. Expected operating conditions for the motors, which are the primary power drain, is operating with a load torque with just the weight of the robot found in the calculation section below (0.371 N·m, 52.5 oz·in). The maximum power draw of each component represents how much power each component can draw at maximum electrical ratings, which for the motors would be the power drawn when the load torque has exceeded the motor’s potential.

Table 9: Power draw of hexapod components

Component	**Typical Power Draw [W]**	**Maximum Power Draw [W]**
Pololu Motors x2	24.0	36.0
Pololu Dual Motor Driver	0.120	0.840
Motor Encoder x2	0.100	0.100
Arduino Uno	0.750	1.00
XBee Module	0.149	0.149
Sum	**25.12**	**38.09**

The sum of the power draw for the hexapod can then be used to find the capacity of the battery required. The equation below takes this power in Watts (W), a factor of safety to account for a difference in actual power draw, and the run time the hexapod is expected to meet.

$$Battery\ Capcity\ [mAh] = Factor\ of\ Safety * \frac{Power\ Draw\ [W]}{Battery\ Voltage\ [V]} * \frac{1000\ mAh}{1\ A} * run\ time$$

Table 10: Power calculations to find battery capacity

	Motor Run Time	Power Draw w/ Motor Consideration [W]	$Battery\ Capcity\ [mAh]$ $1.25 * \frac{Power}{11.1\ V} * \left(\frac{1000\ mA}{1\ A}\right) * 0.5\ hrs$
Typical Power Draw	Half Time (1/2)	13.12	738.8
	Constant (1)	25.12	**1414**
Maximum Power Draw	Half Time (1/2)	20.09	1131
	Constant (1)	38.09	**2144**

Table 10 contains the calculations for finding the necessary battery capacity. Using both the typical and maximum power draw and motors running constantly and for half the operational time yields a range of 1414 mAhrs to 2144 mAhrs. With this information a Lithium-Polymer battery with a capacity of 2200 mAhrs was selected as the power source for the hexapod. This battery chemistry allows for high discharge currents and a long lifespan. The Lithium-Polymer battery used in the prototype is shown in Figure 19.

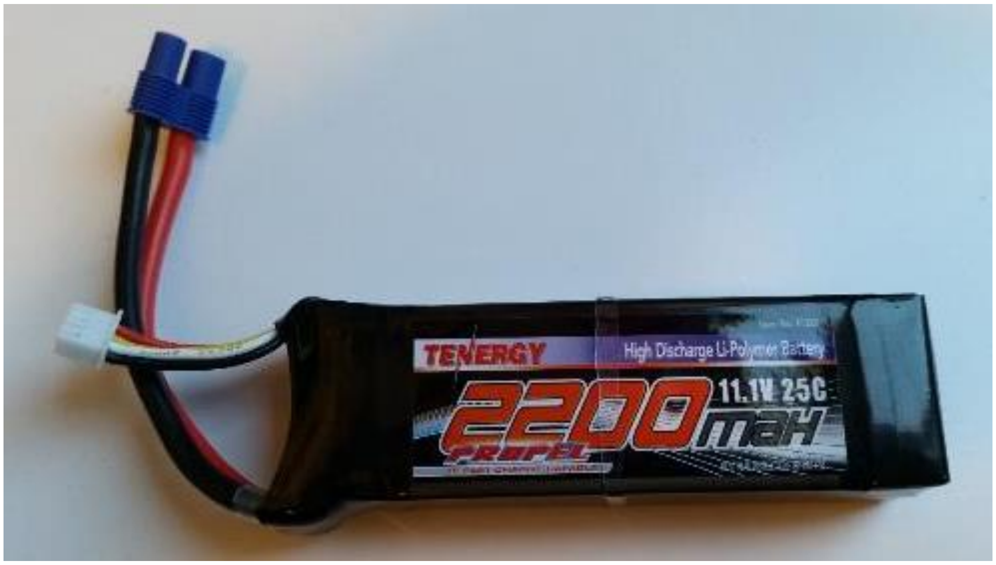


Figure 19: Hexapod battery

Range Testing

The effective range of the XBee modules is given by Digi as 100 m or ~330 ft. This makes the assumption of line-of-sight between the transmitter and receiver which is the ideal situation. Determining actual operational range of the two modules requires experimental testing. Digi's X-CTU software provides a range test function which required one module to be connected to a computer; and a remote device paired to the connected device. The program generates packets of data to be sent and tracks how many are received at the remote module. Also, it generates the received signal strength in dBm and plots it with a success chance as a function of the received signal strength.

Figure 20 shows the range test results when the two XBee modules are stationary on a table, separated by 0.5 m. The green line whose units are on the left side of the graph is the received signal strength of every packet by the transmitter. The blue line indicates the percentage of packets that were received by the receiver and retransmitted to the transmitter. The expected result would be a consistent signal intensity across the test with some variation in strength but consistent connectivity, which is common in wireless communication.

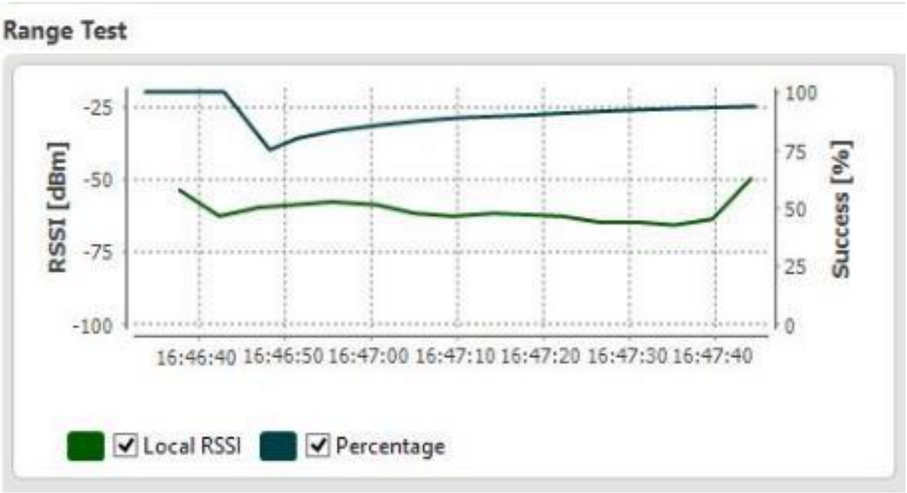


Figure 20: Example X-CTU range test results with 0.5 m separation

This test shows the effectiveness of the link between the modules at a particular range, but not the maximum range possible. Finding the maximum range of these two XBee modules requires moving either the transmitting XBee, which is connected to a computer, or moving the receiver. In this instance it makes more sense to move the receiver by walking away from the transmitter and monitoring the range test display to see where the transmission success chance drops below an acceptable level, or else packets are no longer being received. Figure 21 shows the plot of received signal strength over time while distance is increased on a paved sidewalk. Between 16:56 and 16:57 is when the receiver went out of range in this case. The receiver is moved back into range before 16:58 and this distance was marked as the maximum effective range of our hardware. The distance was measured to be 23.1 m or 75.8 ft.

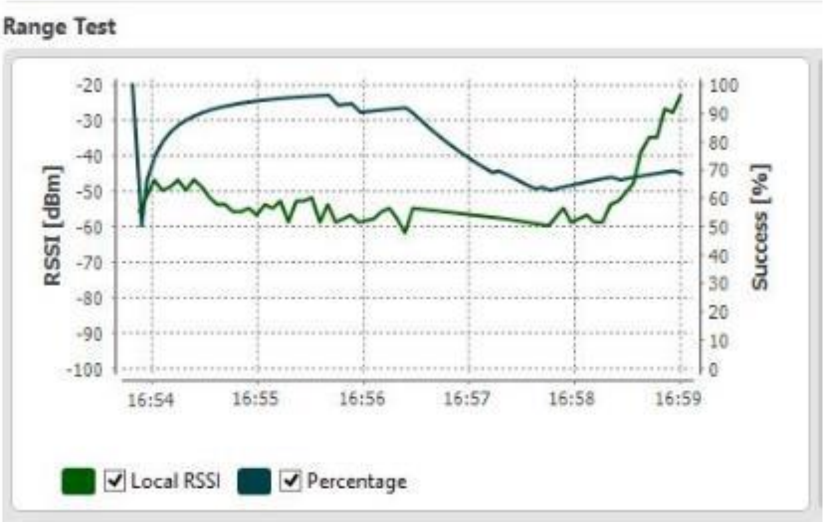


Figure 21: X-CTU range test graphs with modules outside

Leg Orientation Issues

While testing the hexapod's movements a few issues arose with the leg orientation. The primary problem was that the legs were completing more than a full rotation after each command was executed. On initial inspection it appeared that the problem was in the interrupt service routines that were responsible for turning off the motors after a full rotation which is supposed to occur when 900 encoder pulses occurred. Figure 22 illustrates the difference in leg start position versus the location it stops in after one rotation and a second rotation.

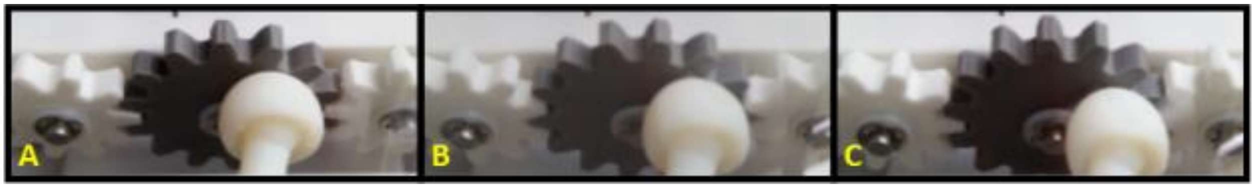


Figure 22: Comparison of (A) leg start location, (B) position after one rotation, and (C) position after two rotations

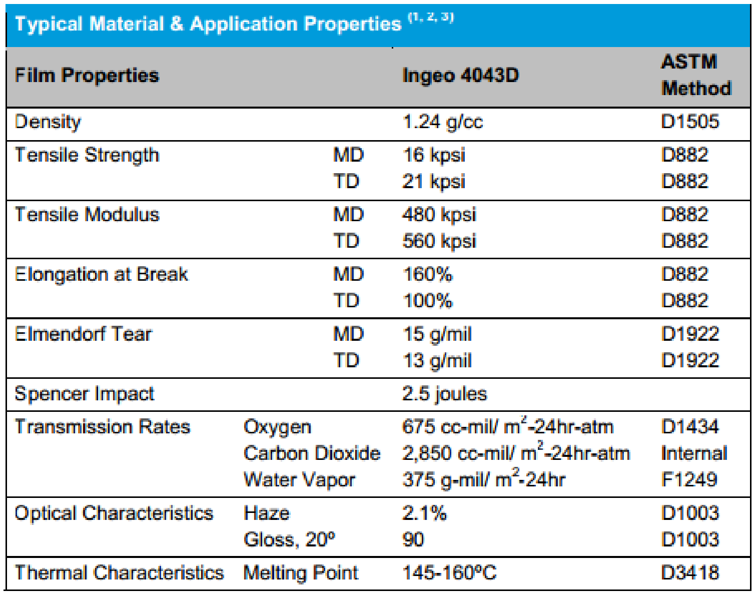
Upon further inspection with different speeds of motor rotation, the problem was seen to actually be stemming from the motor's momentum after the speed was set to zero. Turning off the motors is done by reducing the motor speed to zero, not setting any kind of brake so the motor is still free to turn for a small amount of time before external forces slow the rotation to a stop. Changes were made to the encoder count value that each motor had to reach to count for a full rotation. This was done in order to experimentally reach a value that would cause the legs to stop at one full rotation without going any further. Reducing the encoder counts to 850 brings the offset to a negligible amount.

Materials Research

Materials considered for the prototype were restricted to those which had proven to be effectively used for 3-D printing. The most popular method of low cost 3-D printing utilizes thermoplastic materials, which become soft and malleable when heated but return to a hard solid state when cooled. In order for a thermoplastic to be considered viable for 3-D printing, it must satisfy three basic requirements. First, the material must easily be manipulated into the 3-D printer plastic filament. Second, the material must extrude smoothly and consistently during the application process. Finally, the material must cool and solidify to meet the desired physical characteristics (strength, durability, etc.). Based on availability and cost, the team narrowed potential materials to Polylactide (PLA) and Acrylonitrile Butadiene Styrene (ABS) thermoplastics**.**

Assessing the Viability of 3-D Printing Materials

A primary concern when manufacturing 3-D printed parts made of materials such as PLA or ABS thermoplastics was whether or not the material would be strong enough to withstand the tensile and shear forces which occurred during movement. In order to assess the viability of using 3-D printed materials to construct our prototype, the team completed initial calculations using the strength properties (see Figure 23) of Natureworks Polyactide Resin - Ingeo Biopolymer 4043D, a form of PLA commonly used in 3-D printing applications.



Typical Material & Application Properties (1, 2, 3)

Film Properties		Ingeo 4043D	ASTM Method
Density		1.24 g/cc	D1505
Tensile Strength	MD	16 kpsi	D882
	TD	21 kpsi	D882
Tensile Modulus	MD	480 kpsi	D882
	TD	560 kpsi	D882
Elongation at Break	MD	160%	D882
	TD	100%	D882
Elmendorf Tear	MD	15 g/mil	D1922
	TD	13 g/mil	D1922
Spencer Impact		2.5 joules	
Transmission Rates	Oxygen	675 cc-mil/ m^2-24hr-atm	D1434
	Carbon Dioxide	2,850 cc-mil/ m^2-24hr-atm	Internal
	Water Vapor	375 g-mil/ m^2-24hr	F1249
Optical Characteristics	Haze	2.1%	D1003
	Gloss, 20º	90	D1003
Thermal Characteristics	Melting Point	145-160ºC	D3418

Figure 23: Typical material & application properties of Ingeo Biopolymer 4043D

To get a better understanding of the material and its strength capabilities, the Specific Strength (C) of the PLA was calculated and compared with other building materials using the equation below:

$$C = \frac{\sigma}{\rho}$$

Where σ is the tensile strength of the material and ρ is the material's density. Taking the lowest tensile strength of PLA to be 16 kpsi and converting to Pascal's we calculated the following:

$$\sigma = 16\ kpsi \cdot \frac{1000\ psi}{1\ kpsi} = 16000\ psi$$

$$\sigma = 1600\ psi \cdot \frac{6894.76\ Pa}{1\ psi} = 1.10E8\ Pa$$

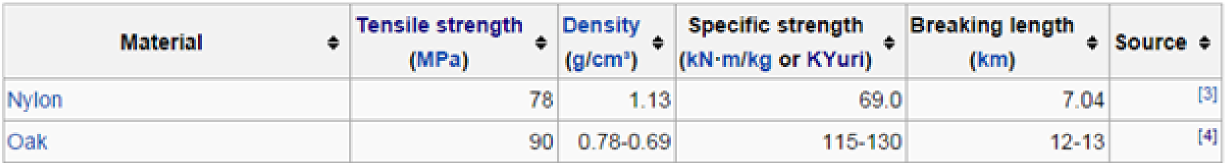
$$\sigma = 1.10E8\ Pa \cdot \frac{1MPa}{1.0E6} = 110\ MPa$$

$$\rho = 1.24\ \frac{g}{cm^3} \cdot \frac{1\ cm^3}{1.0E-6\ m^3} \cdot \frac{1\ Mg}{\mathbf{1.0E6\ g}} = 1.24\ \frac{Mg}{m^3}$$

$$C = \frac{110\ MPa}{1.24\ \frac{Mg}{m^3}} = 88.71 \frac{kN \cdot m}{kg}$$

As can be seen in Table 11, PLA's Specific Strength is comparable to nylon and oak, yet surpasses them both in tensile strength (the tensile strength of PLA is 110 MPa). It is also denser than nylon at 1.24 g/cm^3. This information supported the team's initial prediction that PLA material would adequately withstand the stresses sustained by components during locomotion.

Table 11: Specific tensile strength of various materials



Material	Tensile strength (MPa)	Density (g/cm³)	Specific strength (kN·m/kg or KYuri)	Breaking length (km)	Source
Nylon	78	1.13	69.0	7.04	[3]
Oak	90	0.78-0.69	115-130	12-13	[4]

Heat Testing

The relatively low melting temperatures associated with thermoplastic materials can become problematic when used in assemblies which are subjected to high temperatures. In the case of our hexapod, the heat generated by the motors and on board electronics could quickly compromise the rigidity of the frame if their temperatures approach or exceed the melting temperature of the material used to 3-D print. For this reason, the team conducted a heat experiment to determine the effects of increasing temperature on PLA material. Testing took into account physical characteristics including malleability, melting point, strand deformation, corner strength, and the temperature at which the material is too hot to handle. The materials, procedure, and corresponding results of the test are summarized in Appendix B: Heat Testing.

Rapid Prototyping

Introduction

The term *additive manufacturing* is defined as successive layers of material being added together in a specific axis, in a computer-controlled environment, resulting in the manufacture of a three-dimensional object. In the early 1980's, a commonly used process called stereolithography was created in which the successive layers of the raw material (filament) were added by photopolymers being cured with UV lasers. Shortly afterwards, this process became the most commonly used additive manufacturing process of the 21st century. The file format widely used by 3-D printing software today most commonly known as STL (*ST*ereo*L*ithography) was built from this process and invented by Chuck Hull in 1984. Currently, additive manufacturing technology encompasses various types of processes such as inkjets, polyjets, aerosol jetting, selective laser sintering (SLS) and selective laser melting (SLM), electron bean melting (EBM), stereolithography (SLA), and fused deposition modeling (FDM). The main difference between these processes are the different methods in forming the material to the desired shape. Most commonly available printers use FDM, which melts the material to produce the layers, whereas other processes such as polyjet use liquid curing UV (ultraviolet) lamps. All of these processes can be utilized in various applications such as rapid prototyping, specialized manufacturing, data visualization, and product development (Nathan & Excell, 2010).

It is essential to understand the significant advantages and disadvantages of additive manufacturing over subtractive manufacturing. Subtractive manufacturing is an umbrella term for any process in which a raw material is machined into a desired final shape with the aid of a controlled material removal technique. Modern day subtractive manufacturing is mostly accomplished through the use of CNC (computer numerical control) machining under the supervision of a machine operator and CNC router. The comparison between additive and subtractive manufacturing methods can be achieved by considering various factors such as the complexity, precision, quantity per time and cost of the part being manufactured.

Additive manufacturing processes are able to produce a much more complex geometry and complexity in a given timeframe due to the layer-by-layer build of the material, whereas subtractive methods most often cannot create hollow parts in one piece. Because machining is not entirely automated and consumes human labor, the cost for manufacturing is significantly greater than that of additive methods. Although the material for additive manufacturing is predominantly plastics, new research and innovation shows significant development in printing metals and silicone. Attaining the desired part quality is dependent on the user's preference and need.

For prototyping applications and comparatively short-term use, the most suitable option is additive manufacturing due to the automated nature of rapid prototyping. Subtractive manufacturing, on the other hand, requires previous setup of the CNC machine router by a specialist that has been previously trained on the tooling path of the machine. This project focuses on the preliminary proof of concept of the hexapod, which is in prototype phase. Thus, taking the above factors into consideration, the team focused on the rapid prototyping aspect of additive manufacturing, more commonly known as 3-D printing. Although a number of different 3-D printing tools were used in the creation of the prototype, the team constructed most of the parts using the MakerBot Replicator 2, a relatively low cost system which uses the FDM process.

Printed Parts Overview

To keep track of all 3-D printed parts, the team constructed Table 12 to summarize each parts' printing methodology. Each of the 3-D printed parts can be seen labeled in Figure 24.

Table 12: 3-D printed parts overview. (*): 1 part printed in 4 segments

Part No.	Part Name	3-D Printing:	Quantity
1	Frame	In four parts	1*
2	Universal Joint Ring	As whole	6
3	Leg and ball-socket joint*	As assembly	6
4	Lower Leg	As whole	6
5	Idler Gear	As whole	4
6	Spur Gear	As whole	4
7	Gear Mount	As whole	2
8	Driving Gear	As whole	2
9	Motor Mount	As assembly	1

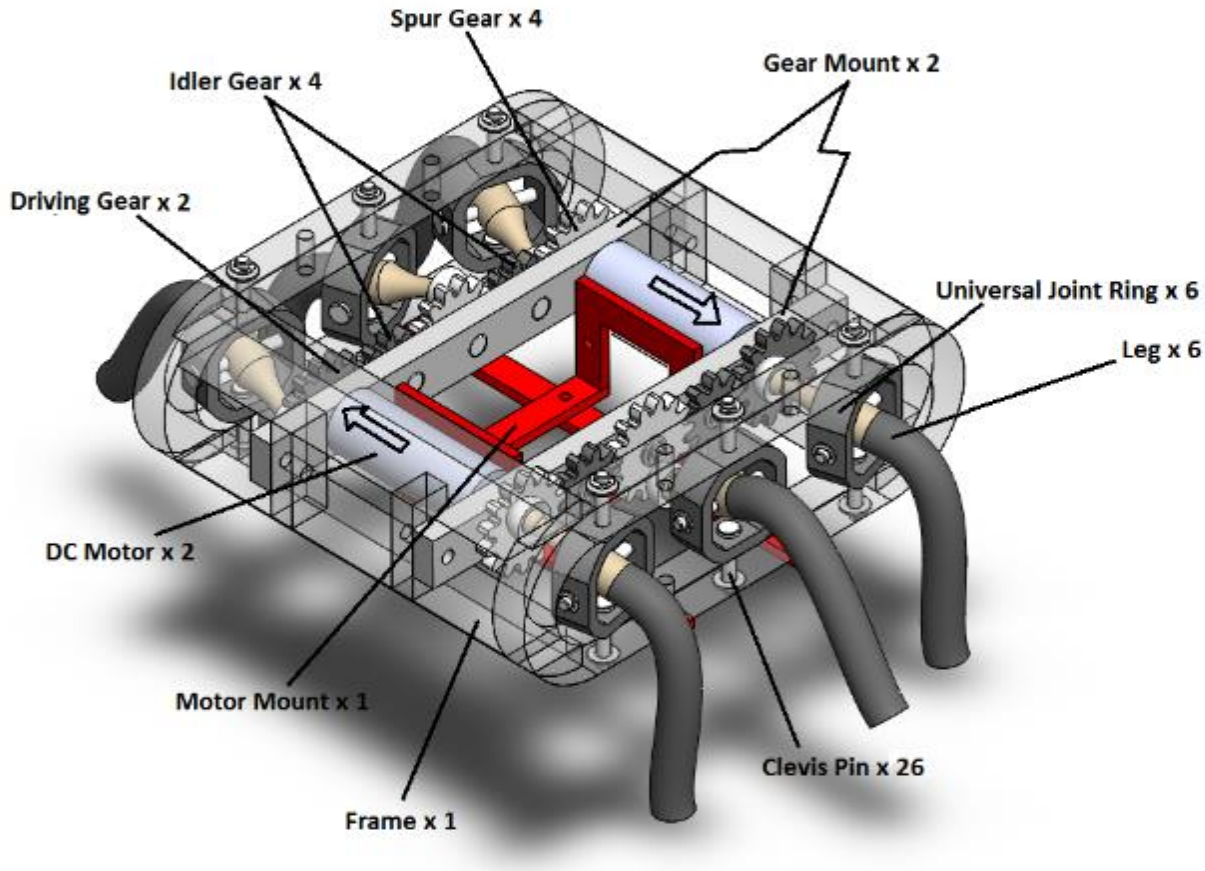


Figure 24: Entire labeled model. Arrows on the motors show where the shaft enters the gear mount and spur gear.

Additive Manufacturing of the Parts

The manufacturing of each part was completed on a case-by-case basis. The majority of the 35 parts were modified slightly from their original SolidWorks designs in order to better fit tolerancing requirements. Some parts were printed with slight tolerancing variations in order to produce an optimal assembly. The following subsections describe the slight modifications done on each different part, procedures followed during their production, and their determined tolerances.

Frame (Part no. 1)

The frame serves the main purpose of a base. All the components of the robot are connected to it, including the gear mount, motor mount, and leg subassemblies. There were two major reasons that played a role in the modification of the frame. The build volume of the available 3-D printer (MakerBot Replicator 2) was 24.6L x 15.2W x 15.5H cm (9.7” x 6.0” x 6.1”). This meant that none of the three dimensions of any part could exceed these dimensions. Our design of the frame had dimensions of 17L x 21W x 7H cm (6.7” x 8.3” x 2.8”), hence one of these dimensions was over the sizing limitations.

Therefore, the first printing method for frame prototyping consisted of printing the part in two pieces then connecting the two parts using acrylic adhesives. Figure 25 shows the two parts being connected.

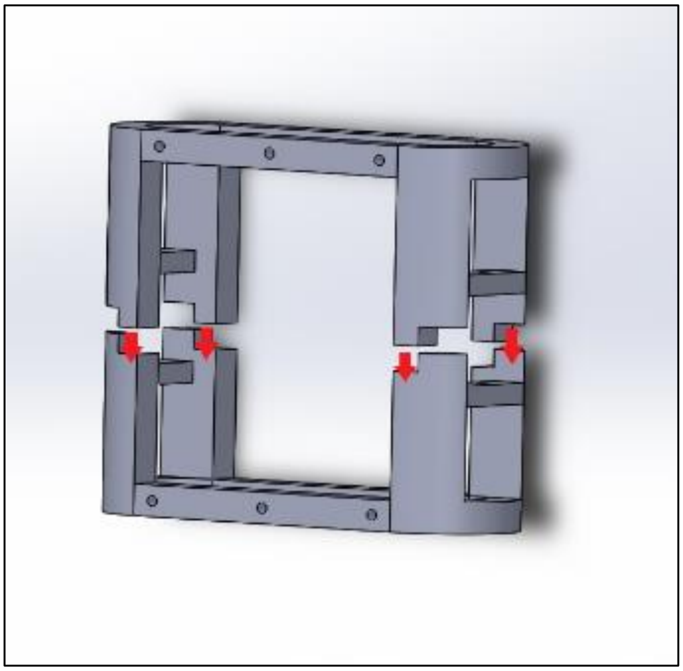


Figure 25: Frame split, printing method #1

The connecting points between the two parts were strengthened using male-female pairing, which created more surface area for the adhesive and supported its hold, reducing any shear forces applied. The second major reason for the alteration was to try and reduce part warping.

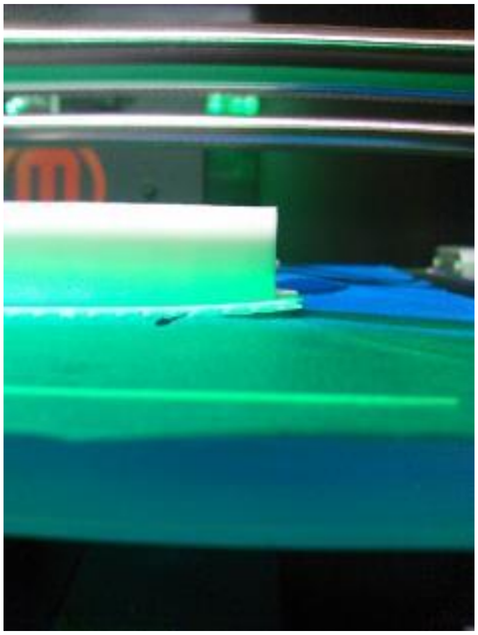


Figure 26: Warping occurring during printing process

Warping, a common occurrence when 3-D printing larger parts, happens when the filament cools too quickly (see Figure 26). As the sequential layers are being printed warping can occur due to the temperature difference between the layers. The shape of the objects can become distorted in a number of ways; the object can be bent, twisted, lifted off of the bed, or succumb to waving. Warping generally becomes a problem in bigger parts with larger surface areas. There are a few possible precautionary measures that can be taken to reduce part warping. Making the surface area of the part smaller, strengthening adhesion between the part and the printer bed, and using isopropyl alcohol to clean the printing surface are all methods used to try and reduce warping. Even after these techniques are utilized, in some parts a small amount of warping is inevitable.

Still concerned about potential warping of the parts, the team produced a second revised printing method to reduce the arising of warping. The second printing method for the manufacturing of the frame consisted of further dividing the frame into even smaller parts. By dividing the frame into four pieces the warping and print time of each section was reduced. Creating a part in less time allowed the team to detect any problems earlier on, rather than at the end of printing the entire frame as one part (this would take much longer to print and use more supporting material). The division of the frame according to the team's second printing method can be seen in Figure 27.

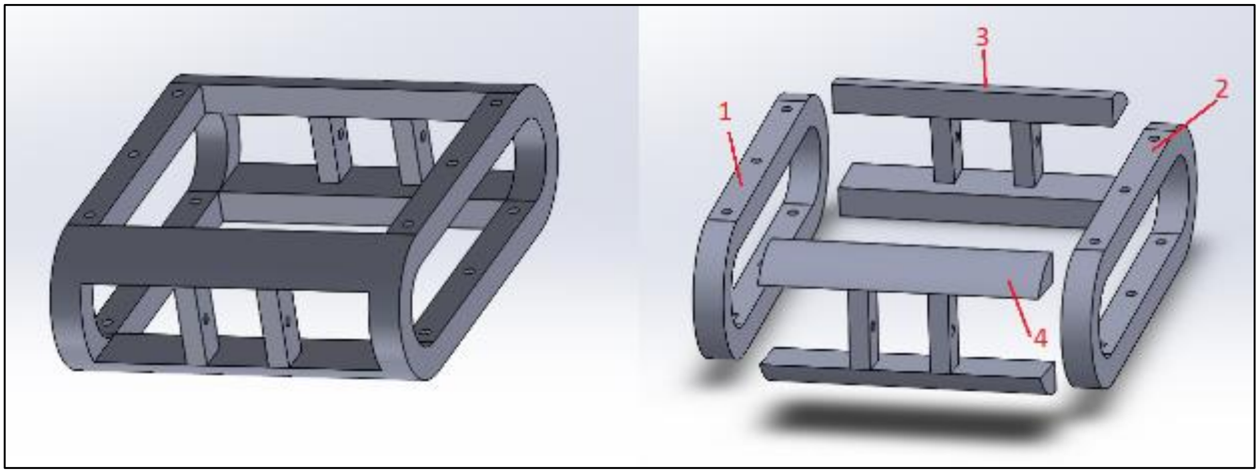


Figure 27: Whole frame before division (left), Frame divided into 4 segments to prevent warping (right)

Slot-shaped segments 1 and 2 were printed running the length, with the layers forming the shortest dimension. The reduction of the height compared to the two-piece option reduced part warpage. Also, this orientation significantly reduced the use of support material which would have been used if it was printed on its side. H-shaped segments 3 and 4 were printed with their flat face down on the print platform.

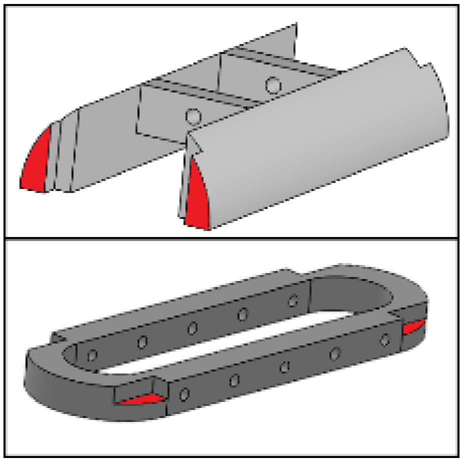


Figure 28: Mating of the frame components

By positioning these pieces with the flat face down, the amount of required support material was minimized. The only support material needed for the printing of the slot-shape and the H-shape was for the printing of the pin holes, which were almost negligible compared to the total volume of the part. Figure 28 shows the H-shaped and slot-shaped segments of the frame, respectively from top to bottom. The red color signifies the faces in contact after attachment of the segments to create the finished frame, male-female coupling was utilized for a strengthened connection.

Universal Joint Rings (Part no. 2)

A universal joint allows motion in the z-axis as well as the x-y plane, housing the legs. The horizontal clevis pins go through the legs, connecting them to the universal joint ring. The universal joint rings are connected to the left and right side of the frame by the help of smaller clevis pins attaching the ring from its upper and lower holes to the slot-shaped piece of the frame. While the horizontal pin allows the leg to move up and down in the z-axis by having the hole-pin connection the pivot, the vertical pins allow the ring to be able to spin left and right causing the legs to move in the x-y plane. In Figure 29: Universal joint to frame connection, the horizontal clevis pins go through the legs and connect to the universal joint rings, and the smaller vertical clevis pins attach the upper and lower wall of the ring to the frame.

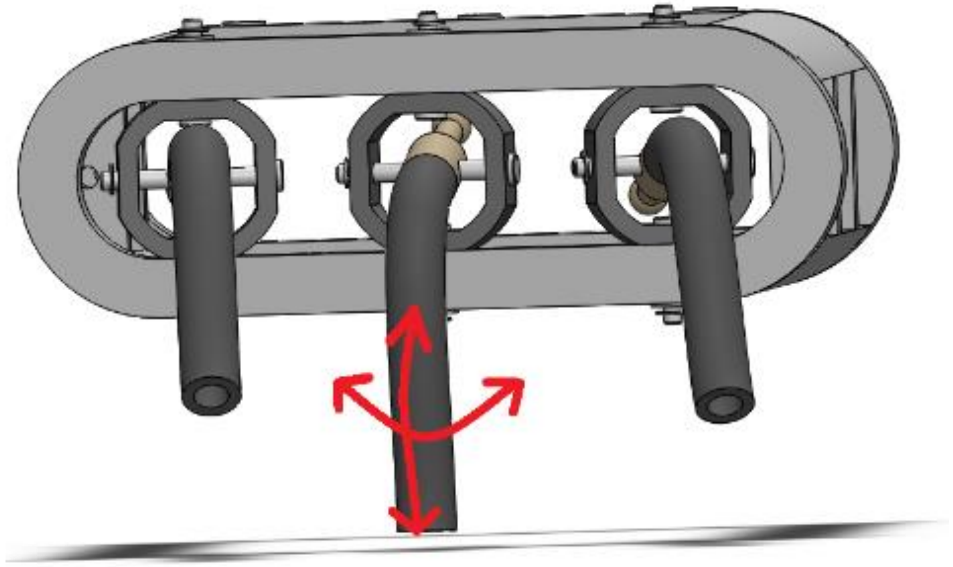


Figure 29: Universal joint to frame connection

The print was done using a horizontal part orientation. For the print, a raft (thin base structure) was used for support and the infill was 10%. The universal joint rings fit between the slot-shaped part (parts 1 and 2 in Figure 27) of the frame. This universal joint converts the rotational motion from the gears to a more linear motion towards the legs. Therefore, the rings twist left and right to accommodate the conversion.

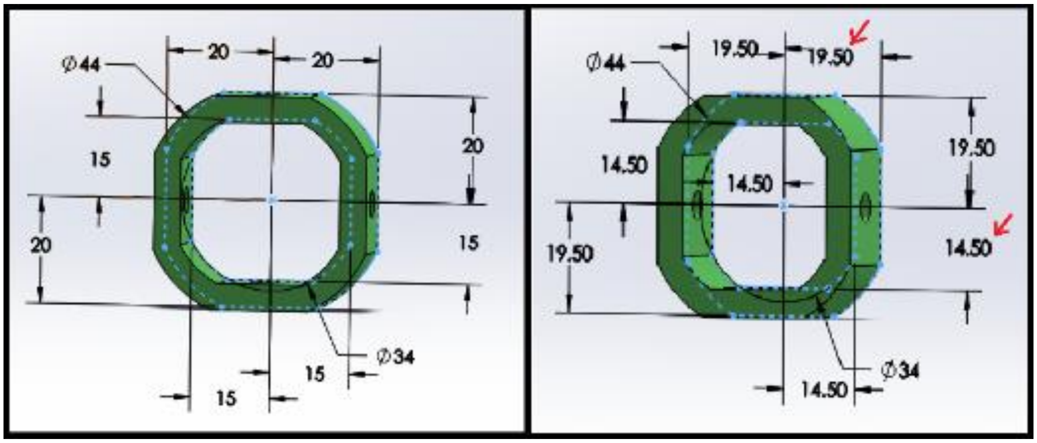


Figure 30: Universal joint ring tolerancing

The universal rings needed to twist comfortably with minimal friction between the inner upper and lower faces of the frame. An appropriate tolerance of -1 mm was given to the x- and y-axes of the universal joint ring for a smooth, yet not-too-loose fit between the sides of the slot-shaped segment of the frame. In Figure 30 above, the left box shows the universal joint ring without the added production tolerance, and the right box shows with the red arrow indicators the added 0.5 mm from the center, totaling a 1 mm overall tolerance.

Ball-Socket Joint (Part no. 3)

Part no. 3 was the most complex part to be printed due to the ball and socket joint attached to the driving gear. The unavailability of the size and orientation of this joint in the market led to the search for alternative manufacturing solutions. Fortunately, more advanced 3-D printers have been able to successfully print assemblies similar to the ball and socket joint. After deciding to 3-D print the part, the team saw two possible ways the part could be manufactured. The first approach was to print the two parts of the joint (ball and socket) with the leg attached as an assembly. The second approach was to 3-D print the two halves of the socket separately, then connect the two halves with the ball in between using adhesion. The second approach was eliminated due to the part's minuscule dimensions and exact precision that would be required when adhering the parts together.

Having very small dimensions (approximate diameter of socket inner sphere: 10 mm [~0.4"]) required great precision, and finer layering than the MakerBot could provide. Therefore, using a printer with finer precision and flexibility became essential. The *Objet 260 Connex* machine has a layer resolution of 16-microns (0.016 mm [0.0006"]), whereas the resolution of *MakerBot Replicator 2* can achieve 100-microns (0.1 mm [0.004"]). Thus, the precision supplied by the Objet machine is over six times that of the MakerBot. The main reason behind the difference in accuracy is the different 3-D printing methods used by each printer. They differ in the technique of processing the raw material.

The Objet 260 uses the polyjet method, which uses liquid cartridge and after each subsequent layer is laid down it uses UV light for curing the liquid into the solid plastic. The MakerBot on the other hand, uses the fused deposition modelling (FDM) method which melts solid filament into a liquid form and lays down the material which eventually solidifies as its temperature returns to room temperature. Due to the use of liquid cartridge, compared to melted solid, the polyjet method is able to achieve a much thinner layer height resulting in increased accuracy. While working with smaller size parts, this precision difference becomes very significant in the structural integrity and print quality. Using the polyjet process would minimize the friction between the ball and socket, which would lead to a longer lifespan and a mechanically sound structure (Fischer, 2008).

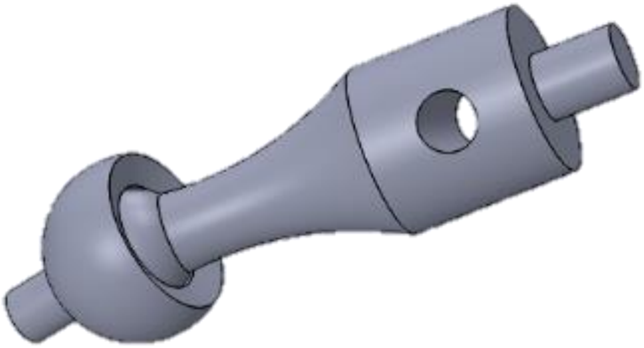


Figure 31: CAD model of the 3-D printed ball and socket joint

Following these considerations, the team used the Objet machine for the printing of part no. 3. Because the “ball” of the joint is part of the leg of the robot, the leg and socket would be printed as an assembly. Upon receiving a quote for the Objet machine, it was evident that printing six assemblies would exceed the team’s budget. This problem was avoided by essentially cutting the leg right after the joint, so that it would be split up into two parts: the ball-socket joint assembly and the lower leg.

When optimizing a CAD design for rapid prototyping, printing tolerance had to be taken into account. According to the Objet machine guidelines, the minimum tolerance of 0.3 mm was required between surfaces in contact during assembly prints (Stults, n.d.). Therefore, this factor was taken into consideration and a tolerance of 0.32 mm was created between the contact of the outer surface of the ball and the inner surface of the socket. Lastly, a small tolerance was added to the hole that houses the horizontal clevis pin connecting the leg to the universal joint ring. A small tolerance of 0.17 mm was given to the connection of the horizontal clevis pin and the pin hole on the ball-socket joint.

Lower Leg (Part no. 4)

The lower leg is the rigid part of the leg that is connected to the ball-socket joint mentioned in the previous subsection. The lower leg is a tubular structure created from a spline sketch, forming the bulk of the leg. The center is hollow in order to reduce material usage and weight. The connection of the two parts is procured by the use of a male-female connection for maximum strength in attachment. The lower leg is the female, the ball-socket joint is the male attachment. The connection was reinforced by adhesion using liquid super glue (Sculpteo, n.d). As a result, the connection of the two parts forming the leg is shown in Figure 32.

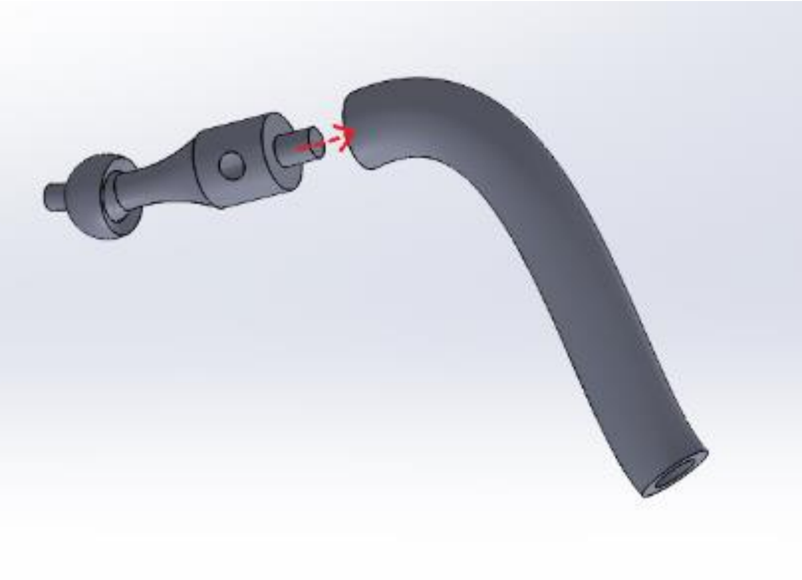


Figure 32: Connection achieved using adhesives, by assembling in the position of the red arrow. The ball-socket joint is on the left, connecting to the lower leg on the right.

The print orientation of the lower leg was horizontal, with the long dimension on the print platform, building up the layers perpendicular to the radius. The print material was PLA plastic, with an infill of 10%. Due to its hollow structure in the middle, support material was used within the leg for reinforcement.

Idler Gear and Spur Gear (Part no. 5 and 6)

The idler gear, which idles between two driving gears controlled by the motor, is the smallest gear in the gear train. The spur gear on the other hand attaches to the ball-socket connection of the leg and hence acts as the direct driving component of the leg's movement. There are two idler and two spurs on the gear train on each side of the robot, making a total number of four idler and four driving gears.

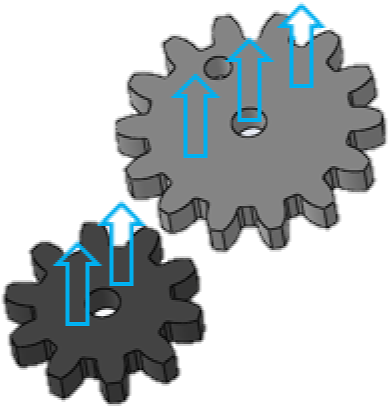


Figure 33: Idler (left) and spur (right) gear print orientation

Both of the gears are connected to the gear mount using clevis pins and nylon washers for extra spacing and support. Figure 33 shows the CAD models of the spur and idler gears. The center holes are for the pins to connect to the gear mount. The second hole closer to the perimeter of the spur gear is the attachment of the socket (ball-socket joint), and subsequently the entire leg.

The parts were printed with the large surface of the gears parallel to the print platform; the print direction is shown with blue arrows in Figure 33. The print material was PLA plastic with an infill of 10%, using a raft for base support. The center holes were added some tolerance and dimensioned in accordance with the clevis pins to ensure a good fit so that the gears were able to rotate with minimal friction around the pins. In order to avoid grinding and wear between the gears, the idler gears were given a tolerance by reducing the pitch diameter by 1 mm. This resulted in a more smooth drive between the gears.

Gear Mount (Part no. 7)

The gear mount serves as the connection hub of the gear train. It is made up of five holes for gear connection, and two end holes for connection with the frame, totaling seven holes on each gear mount. There are two gear mounts in the robot; one on the left side controlling the left legs, and the second is located on the opposite side controlling the right legs. Four of the gear holes are designed to be counter-bore in order to house the head of the clevis pins. This provides a flat inner surface without the pin head extruding, see image (A) in Figure 34. The fifth gear hole is relatively bigger than the other holes because it houses the spur gear, which will be discussed in the next subsection.

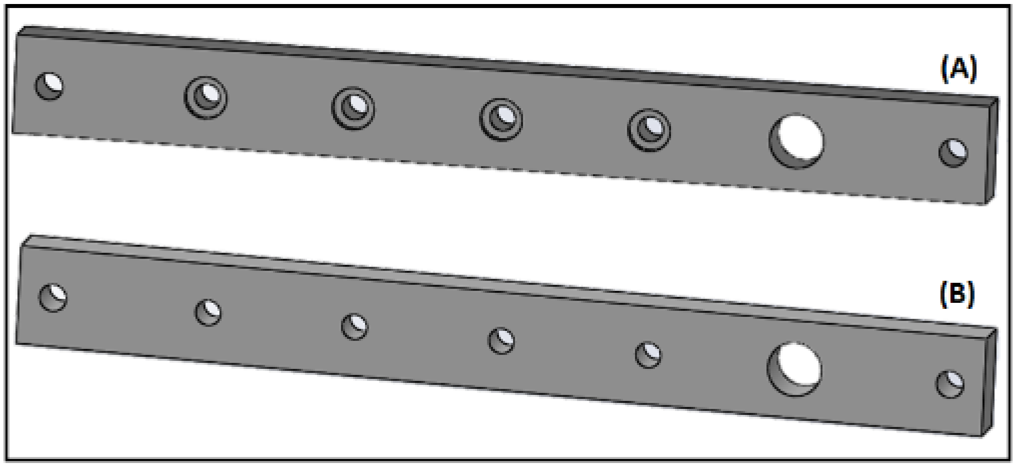


Figure 34: Gear Mount CAD models. (A): inside surface facing the interior of robot. (B): opposite surface facing outwards.

The two gear mounts are connected to the frame in such a way that the surface visible in (A) of both gear mounts face each other through the interior of the robot. The surface in image (B) is where the gears are attached and the legs are connected, which is facing the outside. The print orientation for this part was horizontal, with the longest length on the platform facing up. Rafts were used for support, and the material used was PLA plastic with an infill of 10%. Tolerances of 0.19 mm between the outer surface of the pins and inner surface of the pin holes were used.

Driving Gear (Part no. 8)

Similar to the spur gears and idler gears, the driving gear is a component of the gear train. There is one driving gear on each gear train, totaling to two driving gears overall. The difference between the driving gear and the spur gear is the motor connection shaft of the driving gear. Figure 35 gives a top view of the driving gear, in which the D-profile of the motor shaft can be seen outlined in blue.

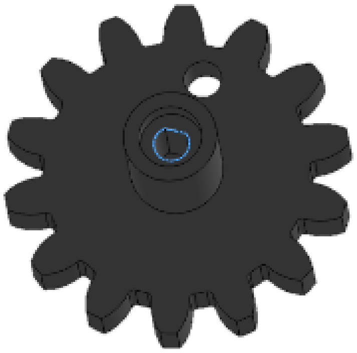


Figure 35: Driving gear with highlighted D-profile hole for motor shaft entry

The main reason that drove us to make it extruded rather than just a normal driving gear with the D-profile outline for a hole, was due to the gear mount being thicker than the length of the motor shaft. Therefore, the shaft was not sufficiently long enough to surpass to the other side of the gear mount to be able to fit into the driving gear. The thickness of the gear mount is 10 mm, and the motor shaft is 8 mm long (see Figure 36, below). Therefore, the letter 'X' on the right box represents the shortage of the motor shaft in order to be able to extend past the gear mount. Additionally, the driving gear having the same thickness as the gear mount did not allow for the shaft to have the extra length to go through the driving gear, which is an additional 5 mm.

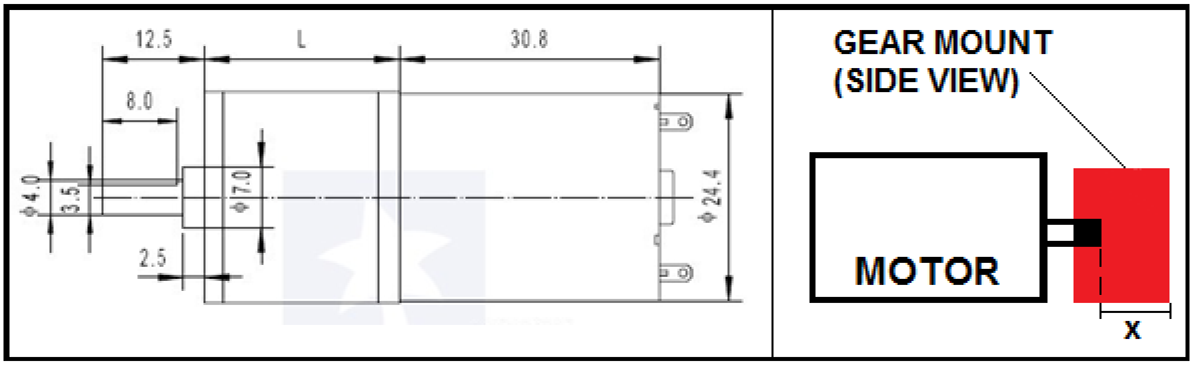


Figure 36: Left: technical drawing of the DC motor – dimensions in mm, motor shaft is 8 mm. Right: diagram of motor connecting to the gear mount, which then connects to the driving gear on the other side. The letter X indicates the shortage of the shaft.

This problem led the team to redesign some components in order to accommodate for the thickness of the gear mount. Making the gear mount thinner was an option, but not an ideal one. If the gear mount was made thinner, warping would increase, causing the shape to bend up and become distorted. Also, if it was less than 10 mm the structural sturdiness would decrease, resulting in insufficient support for the gears and legs. Therefore, the design was altered so that the hole of the motor-gear connection was extruded to accommodate this issue, see Figure 37. With an added tolerance length (0.2 mm), the extruded length was 10.20 mm total. The driving gear meets the motor shaft in the largest hole on the gear mount.

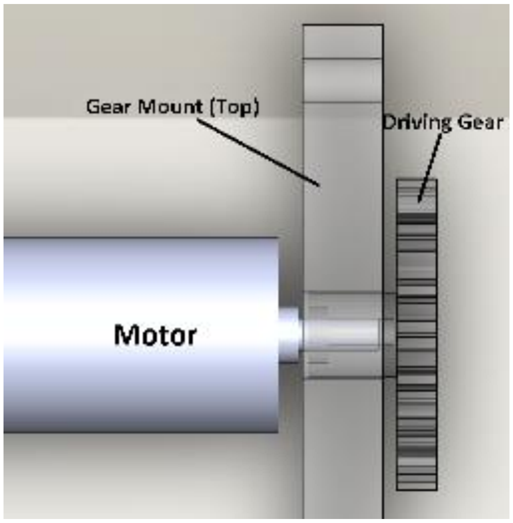


Figure 37: Motor to gear mount - driving gear top view. The motor shaft is in the extruded cylinder, which goes through the largest hole on the gear mount.

Tolerance between the outer surface of the extruded cylinder and the inner surface of the gear mount hole was 0.25 mm, which is almost double the hole-pin tolerances of the other parts. Because the driving gear was to turn in place, it required there to be almost zero friction and sufficient looseness. The print orientation was the same as the spur and idler gears, having the greatest surface area on the print platform and up. Similarly, rafts were used for support and the material was PLA plastic with an infill of 10%.

Motor Mount (Part no. 9)

The motor mount is the part that connects the motors to the frame of the robot. In order to achieve a lightweight design, the team 3-D printed this part instead of using other manufacturing methods. The motors were connected to the motor mount using extra strength Gorilla Tape (see Figure 38).

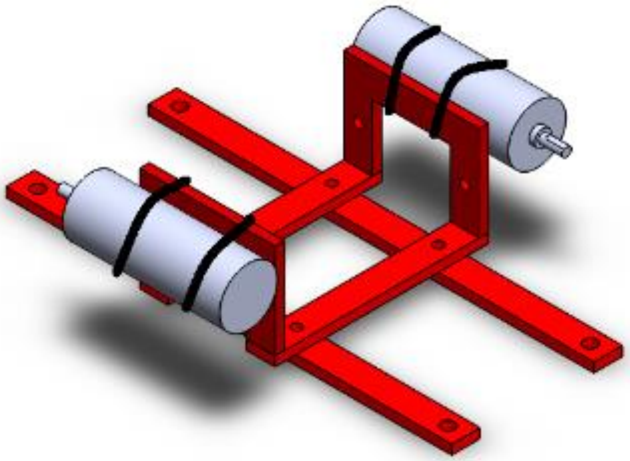


Figure 38: Motor mount (red) attached to the motors using Gorilla Tape (CAD representation)

The motor mount is connected to the bottom of the frame with nuts and bolts. The long end of the motor mount is assembled perpendicular to the sides of the robot allowing the motors to connect to the gear mounts running parallel to the side. The minimal use of materials allowed for maximum spacing in the mid-interior of the robot, see Figure 39 for the motor mount – frame connection.

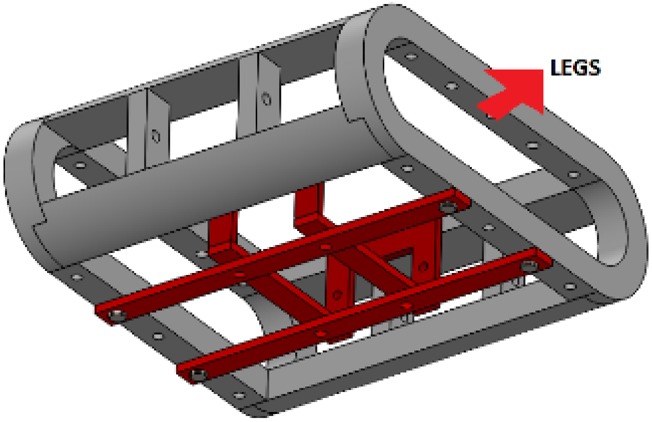


Figure 39: Bottom view of the frame. The red motor mount is connected to the frame from four places using nut-bolt fastening.

The motor mount was oriented as shown in Figure 39 to ensure that the longest dimension was on the platform for printing. Support material was added to the bridge-like structure that the motors connect to, and the print material was ABS plastic.

Conclusion

Overall, the prints were very successful and the team was able to achieve a fully-functional model after assembly. The tolerancing done on the parts formed a good baseline for understanding the different connection types and their tolerance to avoid manufacturing error. Photographs of some of the printed parts can be found in Appendix C: Hexapod Model SolidWorks Parts.

Controller Box

The controller provided us with the directional control of the robot. In order to stay within budget, the team used different rapid prototyping methods for its production. Laser cutting is another rapid prototyping method which allows for the cutting of sheet material under full computer control. Modern laser cutters are able to cut materials such as acrylic, wood, glass, tile, and Teflon to a thickness of 1.5”. This technique is very useful for cutting complex two-dimensional patterns.

The controller box used 2 mm thick acrylic sheet as material, and the maximum dimensions were 126L x 81W x 51H mm (32” x 50” x 20”), see Figure 40 for an exploded view of the rendered CAD model.

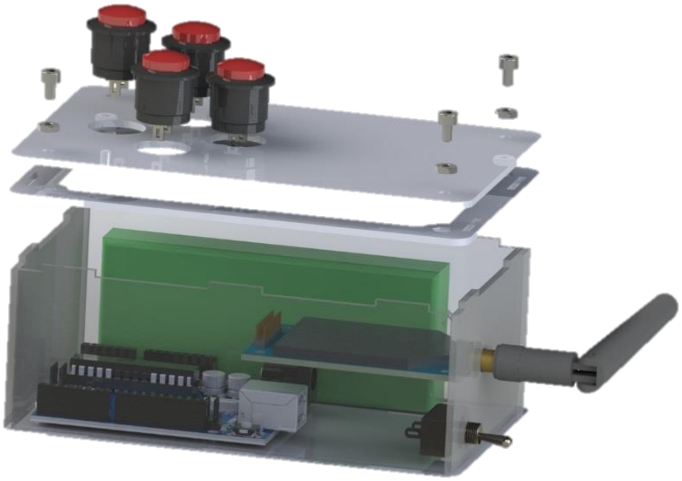


Figure 40: Exploded view of wireless controller (CAD)

The top wall which the buttons are connected to is screwed onto the body using a bilayer design. The second layer (between the top layer and the box) provides a housing for the hex bolts, creating a thread for the screws to put everything together. Each wall is connected to each other using a latch-on system, similar to a puzzle. The male parts connect into the female parts of the cut acrylic parts and are reinforced with super glue. There are two compartments in the box, separating the battery from the rest of the electronics.

The controller is comprised of the following components: an Arduino UNO, an XBee transmitter (with antenna), a Li-Po battery (green casing in the figure), four buttons for direction control, an on/off switch located on the wall below the antenna outlet, and appropriate wiring.

Calculations

The following calculations were used periodically in the electrical and mechanical design process to estimate sustained operational loads, validate design concepts, and select appropriate electrical components for our design.

Expected mass of 3-D printed components

The team used the volume of the finished SolidWorks model (using the SolidWorks Mass Properties too) and the density of PLA to calculate the mass of the model. The density used (1.24 g/cm^3) assumed that parts would be printed with 100% infill, which is unrealistic because parts are typically printed at no more than 10-30% infill. This added a higher factor of safety to the calculations:

$$m = \rho \cdot V$$

$$\rho = 1.24 \ \frac{g}{cm^3}$$

$$V = 691.35 \ cm^3$$

$$m = 1.24 \frac{g}{cm^3} \cdot 691.35 \ cm^3 = 857.27 \ g$$

$$m = 857.27 \ g \cdot \frac{0.001 \ kg}{1 \ g} = 0.857 \ kg$$

Where m is the theoretical mass of the model, ρ is the density of the PLA, and V is the total expected volume of printed material. Using the mass of the model and the acceleration due to gravity we then found the approximate weight of the model (W).

$$W = m \cdot g$$

$$W = 0.857 \ kg \cdot 9.81 \frac{m^2}{sec} = 8.41 \ N$$

Forces due to the weight of the body and additional components

To more accurately find the reaction forces applied due to weight of the robot, the team included the expected weights of on-board electrical components and any fasteners. A breakdown of component mass estimates can be found in Table 13.

Table 13: Expected Mass of Prototype Components

Expected Mass of Prototype Components	
Component Group	Expected Mass
Battery	150 g
Motors	250 g
On-board Arduino and XBee control system	100 g
Mass of 3-D printed components (calculated from model)	857 g
Fasteners	100 g
TOTAL	1457 g

$$1457\ g \cdot \frac{0.001\ kg}{1\ g} = 1.46\ kg$$

Using the total calculated mass of the model and the acceleration due to gravity the team then found the expected weight of the finished robot, including all electrical components and fasteners.

$$W = m \cdot g$$

$$W = 1.46\ kg \cdot 9.81 \frac{m^2}{sec} = 14.32\ N$$

Force acting on each leg due to weight of robot

To find the amount of force acting on each leg at any resting position, we assumed that at any time a minimum of three legs would be touching the ground (consistent with a tripedal gait style) and supporting the weight of the robot.

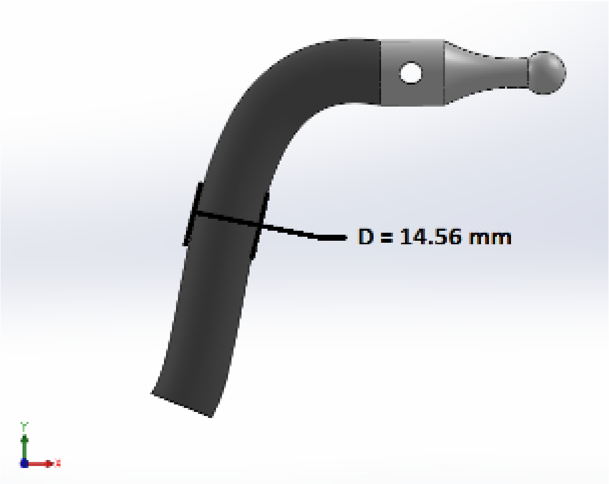


Figure 41: Leg diameter

By using the diameter shown in Figure 41 above we found the circular area of the point of contact between the leg and the ground;

$$A = \pi \cdot r^2$$

Where A is the circular area, calculated using the radius (r) of the leg.

$$r = 14.56\,mm \cdot \frac{1\,m}{1000\,mm} \cdot \frac{1}{2} = 0.00728\,m$$

$$A = \pi \cdot (0.00725\,m)^2 = 1.66E - 4\,m^2$$

The total force per leg was then calculated by dividing the total force acting on the robot by the number of legs in contact with the ground;

$$F = \frac{14.32\,N}{3\,Legs} = 4.77\,\frac{N}{Leg}$$

Finite Element Analysis – Stress Concentrations

After calculating the maximum force applied to the foot of each leg during contact with the ground, the team used Finite Element Analysis tools included in the SolidWorks package to model the maximum expected stress concentrations in each leg during operation of the robot. A von Mises stress simulation was used to determine whether the legs would sustain the maximum expected applied load. The simulation used two fixtures located at the horizontal pinhole and the ball socket head. A force of 4.77 N (taken from the calculations above) was applied to the base of the leg, and the material was set to ABS plastic. The results of the simulation can be seen in Figure 42.

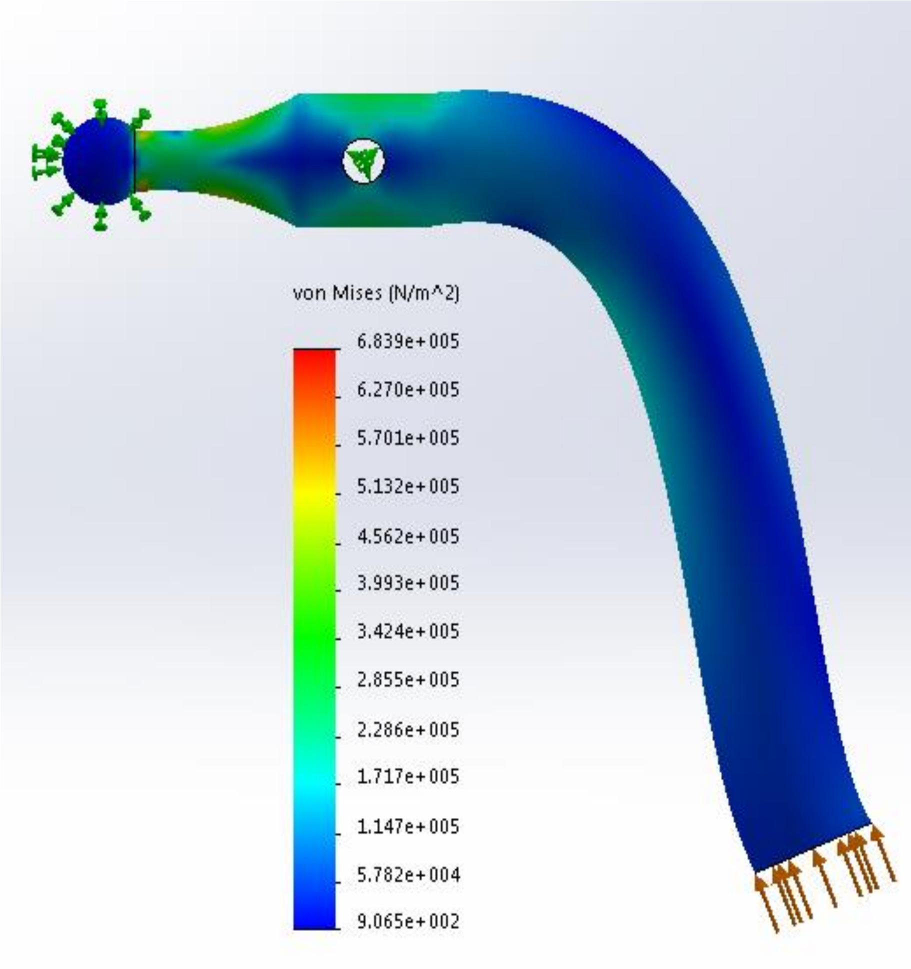


Figure 42: Von Mises stress simulation

Simulation results showed that a maximum stress concentration of approximately 0.7 MPa would be sustained by each leg during motion, located at the neck of the ball portion of the ball-socket. When compared to the yield strength of ABS plastic (approximately 40 MPa), the simulation results suggested that the proposed leg design would be able to withstand the maximum expected forces and experience negligible deformation as a result of the applied load.

Reaction forces due to weight of the robot

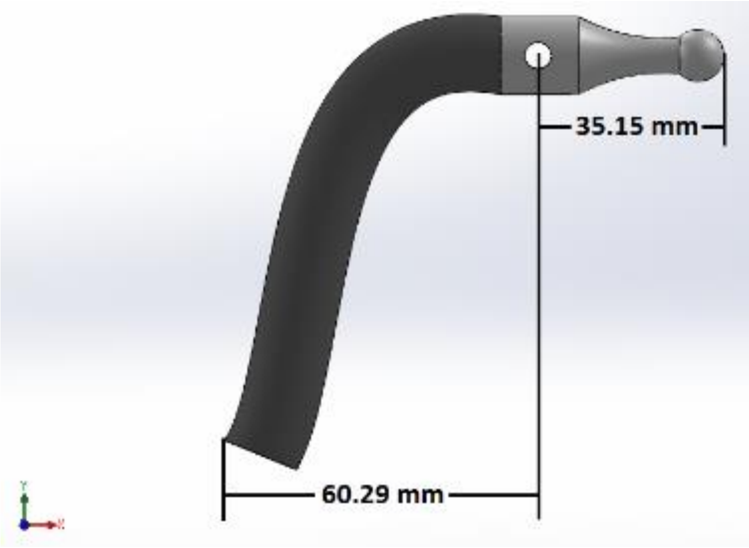


Figure 43: Inner and outer leg dimensions

Using the dimensions of the inner and outer leg found in Figure 43 and the 4.77 N force acting on each leg found in the "force from weight acting on each leg" calculations section, we then calculated the reaction force acting on the inner leg and driving gear.

$$-F_r = F_o \cdot d_1/d_2$$

Where F_r is the reaction force from F_0 (the force acting on the leg), d_1 is the distance from the lever point to F_0, and d_2 is the distance from F_r to the lever point.

$$F_o = 4.77\frac{N}{Leg}$$

$$d_1 = 60.29\ mm \cdot \frac{1\ m}{1000\ mm} = 0.060\ m$$

$$d_2 = 35.15\ mm \cdot \frac{1\ m}{1000\ mm} = 0.035\ m$$

$$F_r = \frac{4.77\ N \cdot 0.060\ m}{0.035\ m} = 8.18\ N$$

Thus, a maximum of 8.18 N of force acts on each inner leg/wheel in an upwards direction, corresponding to when the leg comes in contact with the ground.

Frictional Force (Static)

Next we calculated the frictional forces acting on each leg during motion. Using the equation $F_r = \mu \cdot N$ and the force found in the "reaction forces due to weight of the robot" calculation section, we calculated the following:

$$F_r = \mu \cdot N$$

$$\mu = 0.16 \pm\ 0.02\ (we\ will\ use\ 0.18)$$

$$N = 8.18\ N\ (from\ section\ 3)$$

$$F_r = 0.18 \cdot 8.18\ N$$

$$F_r = 1.47\ N$$

$$Total\ Static\ Force = 8.18\ N + 1.47\ N$$

$$Total\ Static\ Force = 9.65\ N$$

Frictional Force to Overcome (Dynamic)

During operation of the robot its legs are consistently moving, thus enduring a friction force associated with the movement of the gear train. We used the equation $F_r = \mu \cdot N$ and the force found in the section named "reaction forces due to weight of the robot" to calculate the friction force:

$$F_r = \mu \cdot N$$

$$\mu = 0.10 \pm 0.01\ (we\ will\ use\ 0.11)$$

$$N = 8.18\ N\ (from\ section\ 3)$$

$$F_r = 0.11 \cdot 8.18\ N$$

$$F_r = 0.90\ N$$

$$Total\ Dynamic\ Force = 8.18\ N + 0.90\ N$$

$$Total\ Dynamic\ Force = 9.08\ N$$

Torque Required to Overcome the Force from Weight

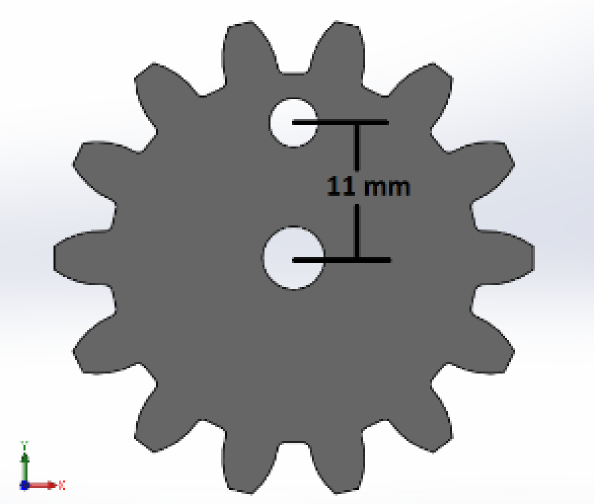


Figure 44: Rotating Gear Radius

To find the torque (τ) required to rotate the gear, we took the calculated static friction force and multiply by the radius of the gear found in Figure 44;

$$\tau = F \cdot r$$

$$F = 9.65\ N$$

$$r = 11\ mm \cdot \frac{1\ m}{1000\ mm} = 0.011\ m$$

$$\tau = 9.65\ N \cdot 0.011\ m = 0.106 N \cdot m/leg$$

To find the amount of torque the motor will had overcome, the team multiplied 0.106 N·m/leg by 2, as there are 2 legs touching the ground per motor.

$$\tau = 0.106\ N \cdot \frac{m}{leg} \cdot 2\ legs = 0.212\ N \cdot m$$

Factor of Safety of the Motor

Using the torque required to overcome frictional forces from the “torque required to overcome the force from weight” calculation section, and desiring a factor of safety of 1.75, we found the torque that the motor needed to meet our expectations:

$$Factor\ of\ Safety = \frac{Expected}{Actual}$$

$$1.75 = \frac{Expected}{0.212\ N \cdot m}$$

$$Expected = 1.75 \cdot 0.212\ N \cdot m$$

$$Expected = \ 0.371\ N \cdot m$$

Motor Selection

Because commercial specifications for motors are given in $\mathrm{oz} \cdot \mathrm{in}$, the group had to convert the motor torque:

$$0.371\ N \cdot m = 52.5\ oz \cdot in$$

Using the above Load Torque and the following equation,

$$loaded\ RPMs\ = load\ torque \cdot \left(\frac{unloaded\ RPMs}{stall\ torque}\right) - unloaded\ RPMs$$

The group created a table comparing various motor specifications (see Appendix D: Gear Motor Comparison). The table helped in comparing and selecting the appropriate motor for our application.

Through comparing the loaded RPMs found in the Gear-Motor Comparisons sheet found in Appendix D, two possible motor speeds were selected and compared based on the distance the robot would travel when the legs were not touching the ground. The calculations were completed for both 77 RPMs, and 25 RPMs for comparison.

To find the linear velocity of the motor we used the equation:

$$V = r \cdot \omega$$

$$r = 0.011\ m$$

$$\omega = 77\ rpm \cdot \frac{2\pi}{60} = 8.06 \frac{radians}{second}$$

$$V = 0.011\ m \cdot 8.06\ \frac{radians}{second} = 0.089\ m/s$$

This velocity was then used to find the time the robot is not touching the ground. We then needed to find the circumference of the wheel.

$$2 \cdot \pi \cdot r = Circumference$$

$$Circumference = 2 \cdot \pi \cdot 0.011\, m$$

$$Circumference = 0.06912\, m$$

Due to the robot not touching the ground half the time, (between 135° to 225° and -45° to 45°) the circumference was divided by 4 to get one of those distances.

$$\frac{Circumference}{4} = \frac{0.6912\, m}{4} = 0.0173\, m$$

$$Time = \frac{distance}{rate}$$

$$Time_{touching} = \frac{0.0173\, m}{0.089} \cdot \frac{1}{2} = 0.097\, s$$

The above time information was used to calculate the distance the robot travels downward in this time.

$$Distance = \frac{1}{2} \cdot a \cdot t^2$$

$$Distance = \frac{1}{2} \cdot 9.81 \cdot 0.097^2 = 0.046\, m$$

These calculations were repeated using a speed of 25 RPM and constant circumference:

$$V = r \cdot \omega$$

$$r = 0.011\, m$$

$$\omega = 25\, rpm \cdot \frac{2\pi}{60} = 2.62 \frac{radians}{second}$$

$$V = 0.011\, m \cdot 2.62 = 0.029\, m/s$$

$$Time = \frac{distance}{rate}$$

$$Time_{touching} = \frac{0.0173\, m}{0.029} \cdot \frac{1}{2} = 0.299\, s$$

$$Distance = \frac{1}{2} \cdot a \cdot t^2$$

$$Distance = \frac{1}{2} \cdot 9.81 \cdot 0.299^2 = 0.44\, m$$

Comparing these two distances, the team concluded that the 77 RPM motor is the best motor choice for this application because the 25 RPM motor will not provide the necessary walking speed required to keep the robot from tipping forward.

Conclusions

Prototype Assessment

Testing of the final prototype confirmed that the group was successful in completing the four main objectives of the hexapod project. The side and front views of the finished prototype can be seen in Figure 45 and Figure 46, respectively.

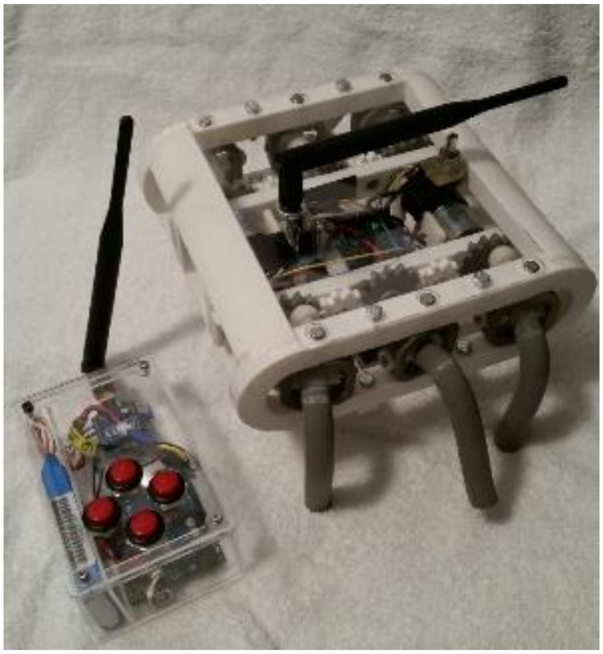


Figure 45: Side view of finished prototype

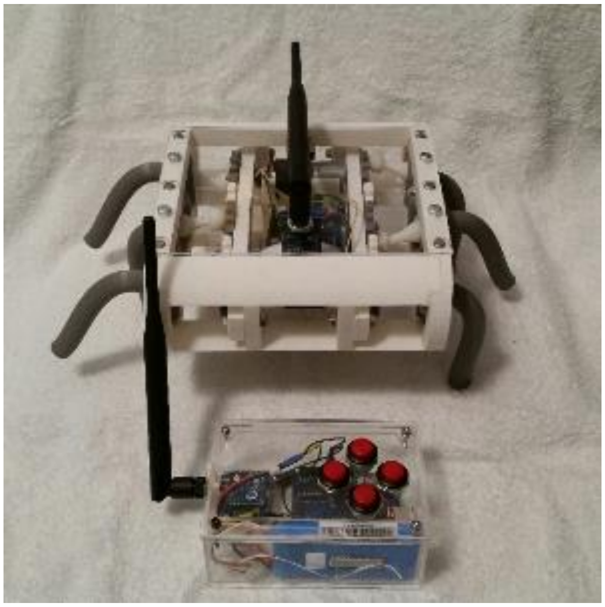


Figure 46: Front view of finished prototype

All design specifications were successfully translated to the first generation prototype. The materials used for prototype production (PLA and ABS plastics) were lightweight and durable, yet easily replicable. In the event that a part were to break, a replacement part can easily be printed. The prototype is controlled wirelessly, and successfully walked forwards, backwards, and turned. The actuation of all six legs was achieved using only two motors, and provided for a simple method of locomotion.

An additional feature of the hexapod prototype design was its ability to change gait styles without the requirement for additional components. By removing the legs and driving gears of the robot, the gait style could be altered by changing the orientation of the driving gears. Figure 47 depicts two possible orientations of the driving gears which produce two unique gait styles.

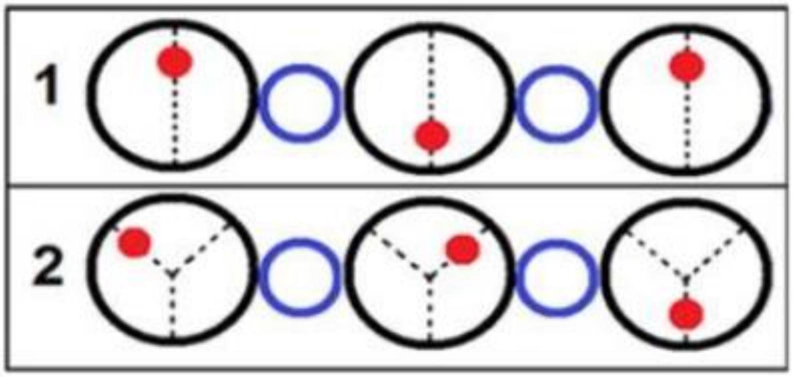


Figure 47: Diagram of multiple gait styles

The black circles represent the driving gears and the blue circles represent the idler gears. The red dots depict the location of the driving gear ball-socket joint. The top diagram shows the leg orientation of a standard tripedal gait system, while the second diagram represents a centipede-like style of motion where only two legs touch the ground at a time.

Future Work

The possibilities for alternative gait styles give a unique opportunity for future work on the hexapod robot. Further testing of gait styles (like the second example in Figure 47) could allow for faster motion styles. Alternatively, future work could also focus on translating the current terrestrial design to aquatic environments. Areas for future work in electrical design include the addition of an on-board camera for the robot or sensors to provide real time data on environmental conditions.

Work Cited

Benyus, J. M. (1997). *Biomimicry* (p. 1). New York: William Morrow.

Birkmeyer, P., Peterson, K., & Fearing, R. (n.d.). DASH: A Dynamic 16g Hexapedal Robot. Retrieved December 12, 2014, from http://robotics.eecs.berkeley.edu/~ronf/PAPERS/IROS09_DASH.pdf

Ding, X., Wang, Z., Rovetta, A., & Zhu, J. M. (2010). Locomotion analysis of hexapod robot. *Climbing and Walking Robots*, 291-310.

Excell, J., & Nathan, S. (2010, May 24). The Rise of Additive Manufacturing. Retrieved December 12, 2014, from

http://www.theengineer.co.uk/in-depth/the-big-story/the-rise-of-additive-manufacturing/1002560.article

Fearing, R., Steltz, E., & Hoover, A. (2008). RoACH: An autonomous 2.4g crawling hexapod robot. *IEEE/RSJ International Conference on Intelligent Robots and Systems,* 26-33.

Fischer, F. (2008, January 1). FDM and Polyjet 3-D Printing. *Stratys Co, from*

http://www.stratasys.com/resources/~/media/722E026DFF2F429C9DA6362BDD42ED2D.pdf

Green, D. B., & Obaidat, A. S. (2002). An accurate line of sight propagation performance model for ad-hoc 802.11 wireless LAN (WLAN) devices. *, 5* 3424-3428 vol.5.

Meet Dash | Dash Robotics. (n.d.). Retrieved December 12, 2014, from http://dashrobotics.com/pages/dash-at-home

Pahlavan, K., & Krishnamurthy, P. (2002).*Principles of Wireless Networks: A unified Approach*. New Jersey: Prentice Hall.

RHex All-Terrain Robot, Boston Dynamics, Waltham, MA.

Sculpteo. (n.d.). How to design a model compatible with 3-D printing? *3-D Printing Tutorial*.

Stultz, E. (n.d.). Objet Material Information. *WPI Rapid Prototyping Guidelines, from*

https://sharepoint.wpi.edu/academics/ME-PROTO/documents/Objet%20Material%20Information%20-%2001-22-2014.pdf

XBee® Wi-Fi RF Module – S6B Manual, Digi International, Minnetonka, MN, Firmware version: 202x, 2014.

Appendices

Appendix A: Program Code

Hexapod Code

```
/* Hexapod v.06*/
/* This sketch controls the functionality of the hexapod. It handles the
   receiving of commands over software serial pins, processing of these
   commands into usable values for the motor driver, and the handling
   of feedback from the motors to aid in control and protection against
   stall and over-current situations*/

//external libraries
#define NO_PORTD_PINCHANGES//disables part of PinChangeInt library
#define NO_PORTB_PINCHANGES//necessary to use software serial library
#include <PinChangeInt.h>//allows for more than default interrupt pins
#include <DualVNH5019MotorShield.h>//provided by Pololu to drive motors
#include <XbeeSoftwareSerial.h>//allows serial comms on non-default pins

DualVNH5019MotorShield md;//allows the use of motor driver commands

//pin declarations
const int encoderM1A = A0; //pin A0 is encoder M1A
const int encoderM2A = A1; //pin A1 is encoder M2A
//SoftwareSerial declaration
XbeeSoftwareSerial XbeeSerial =  XbeeSoftwareSerial(3,5); //Rx, Tx
/* Variables */
//volatile variables to be used in the Interrupt Service Routines
volatile int count1A = 0; //encoder count for motor 1
volatile int count2A = 0; //encoder count for motor 2
volatile boolean M1flag = false;//indicate if motors are on(true) or
off(false)
volatile boolean M2flag = false;
char charBuff[6]; // variable to store incoming serial data
//variables to store data about the received command
byte caseVal = 0; //caseVal refers to the Forward, Back, Left, and Right
commands
byte command = 0; //initialize the received command to 0(do nothing)
int motorSpeed = 150; //Motor Speed

// set-up that will run once upon powering up the Arduino
void setup(){
  Serial.begin(9600);  //baud rate of 9600
  XbeeSerial.begin(9600); //open software serial communication
  md.init();  //initialize pin declarations for motor driver shield
  pinMode(encoderM1A, INPUT);  //set encoder M1A pin as input
  pinMode(encoderM2A, INPUT);  //set encoder M2A pin as input
  digitalWrite(encoderM1A, HIGH);  //set internal pull-up resistor for M1
interrupt
  digitalWrite(encoderM2A, HIGH);  //set internal pull-up resistor for M2
interrupt
  PCintPort::attachInterrupt(encoderM1A, M1count, RISING);  //attach
interrupt for M1
  PCintPort::attachInterrupt(encoderM2A, M2count, RISING);  //attach
interrupt for M2
```

```
  /*initialize the serial buffer to be an array of null characters buffer
should
    be initialized to nothing on start-up but this guarantees a blank
    buffer after set-up */
  for(int i=0; i<7; i++){
   charBuff[i] = '\0';
  }
  delay(1000);  //time (of 1sec) for set-up to take hold
}

// Loop of main program that will run continuously
void loop(){
  if(XbeeSerial.available()>=5){//if serial data is available, get the data
  XbeeSerial.readBytes(charBuff,6); //reads data into the array charBuff
  command = getCommand(charBuff);  //translate received buffer into a viable
cmd
  doMove(command);  //do something based on command
  caseVal = 0; //reset caseVal to 0 to prevent using the same command again
  count1A = 0; //reset encoder variable count after moving
  count2A = 0; //reset encoder variable count after moving
  }//end of if(Serial.available()>0)
}//end of loop

/* getCommand FUNCTION
  Compares received string to known characters. If the known commands, which
are one character in length, are found in the string the corresponding value
is returned by the function.*/
int getCommand(char cmdString[6]){
  String objString = cmdString; //turn the input string into an object
  for(int i=0;i<7;i++){//look at every char in the object
    if ((objString.charAt(i)) == 'F'){//if there is an F cmd is forward
      caseVal = 1;}
    else if((objString.charAt(i)) == 'b'){//if there is an b cmd is back
      caseVal = 2;}
    else if((objString.charAt(i)) == 'R'){//if there is an R cmd is right
      caseVal = 3;}
    else if((objString.charAt(i)) == 'l'){//if there is an l cmd is left
      caseVal = 4;}
  }//end of check for characters
  return caseVal;
  for(int i=0; i<7; i++){//clear the buffer
    charBuff[i] = '\0';
  }
}

/* doMove FUNCTION
   Take in a value between 1 and 4 corresponding to the F,b,l,R options from
getCommand.
   This function outputs nothing, but drives the motors in the necessary
direction and
   speed to achieve the desired movement options. Motor encoder values
determine how
   long the motors run to maintain the correct leg orientation
   */
void doMove (byte command){
  switch(command){//command determines which case runs
```

```
    default:{  // Do nothing, motor speeds set to 0
      md.setSpeeds(0,0);  //m1speed, m2speed
      M1flag = false;
      M2flag = false;
      break;
    }
    case 1:{  //Forward
        md.setM1Speed(motorSpeed);
        md.setM2Speed(motorSpeed);
        M1flag = true;
        M2flag = true;
        stopIfFault();
        break;
    }
    case 2:{  //back
        md.setM1Speed(-motorSpeed);
        md.setM2Speed(-motorSpeed);
        M1flag = true;
        M2flag = true;
        stopIfFault();
        break;
      }
    case 3:{  //Right
        md.setM1Speed(motorSpeed);
        md.setM2Speed(-motorSpeed);
        M1flag = true;
        M2flag = true;
        stopIfFault();
        break;
    }
    case 4:{  //left
        md.setM1Speed(-motorSpeed);
        md.setM2Speed(motorSpeed);
        M1flag = true;
        M2flag = true;
        stopIfFault();
        break;
    }
    while(M1flag || M2flag){
      //while the motors have not rotated fully, wait
    };
    command = 0; //reset the command variable to 0.
    }//end of case statement
}//end of doMove

void stopIfFault(){//adapted from Pololu example
  if (md.getM1Fault()){
    md.setM1Speed(0); //set speed of M1 to 0
    M1flag = false;
  }
  if (md.getM2Fault()){
    md.setM2Speed(0); //set speed of M2 to 0
    M2flag = false;
  }
}

// Interrupt Service Routines (ISR)
// M1count increments every time pin A0 gets a rising edge. (M1A encoder)
void M1count(){
  count1A++;//increment count
  if(count1A >= 765){//if count corresponds to a full rotation
    md.setM1Speed(0);//stop the motor
    M1flag = false;//reset the flag
  }
}

//M2count increments every time pin A1 gets a rising edge (M2A encoder)
void M2count(){
  count2A++;
  if(count2A >= 765){
    md.setM2Speed(0);
    M2flag = false;
  }
}
```

Controller Code

```
/*Controller v.11*/
/*This sketch handles the polling of pin changes to determine appropriate
commands that should be sent to the hexapod over the XBee modules.*/

//declare pins as constants. Four pins for the four different btn cmds.
int forwardPin = 13;
int rightPin = 12;
int backPin = 11;
int leftPin = 10;
//set commands to be constant character strings
const char forward[6] = "?FFF>";
const char back[6] = "?bbb>";
const char left[6] = "?lll>";
const char right[6] = "?RRR>";

int val = 0;// case variable

void setup(){
  //initialize serial
  Serial.begin(9600);  //baud rate of 9600
  //set pin declarations
  pinMode(forwardPin, INPUT_PULLUP);  //digital cmd pins
  pinMode(rightPin, INPUT_PULLUP);  //set internal pullup resistors
  pinMode(backPin, INPUT_PULLUP);  //prevents pins from "floating" between
0&5V
  pinMode(leftPin, INPUT_PULLUP);
  delay(3000);  //time for setup to take hold
}

void loop(){
  //begin polling for changes to the 4 input pins.
  delay(875);
  if(digitalRead(forwardPin) == LOW) {
    val = 1;}
    else if(digitalRead(backPin) == LOW){
      val = 2;}
      else if(digitalRead(rightPin) == LOW){
        val = 3;}
        else if(digitalRead(leftPin) == LOW){
          val = 4;}
          else {
          val = 0;}
  //switch case determines serial data to be sent
  //val1=fwd, val2=bck, val3=right, val4=left
  switch(val){
    case 0://nothing happens
      val = 0;
      delay(100);
      break;
    case 1:
      Serial.write(forward);  //forward
      delay(100);
      val = 0;
      break;
    case 2:
      Serial.write(back);  //back
      delay(100);
      val = 0;
      break;
    case 3:
      Serial.write(right);  //right
      delay(100);
      val = 0;
      break;
    case 4:
      Serial.write(left);  //left
      delay(100);
      val = 0;
      break;
    default:    //default should write nothing
      val = 0;
      delay(100);
      break;
  }
}
```

Appendix B: Heat Testing

Materials:

- Aluminum Foil Baseplate
- Sample piece of PLA
- Safety Gloves
- Variable Temperature Oven

Procedure:

Step 1: Preheat oven to desired temperature

Step 2: Place PLA sample on aluminum foil baseplate

Step 3: Place sample and baseplate in oven

Step 4: Wait ten seconds then begin three minute timer

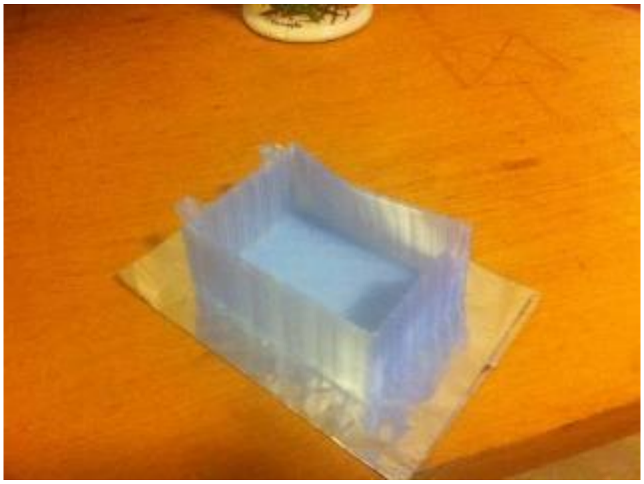
Step 5: After three minutes, remove sample and baseplate from oven.

Results:

At 170°F

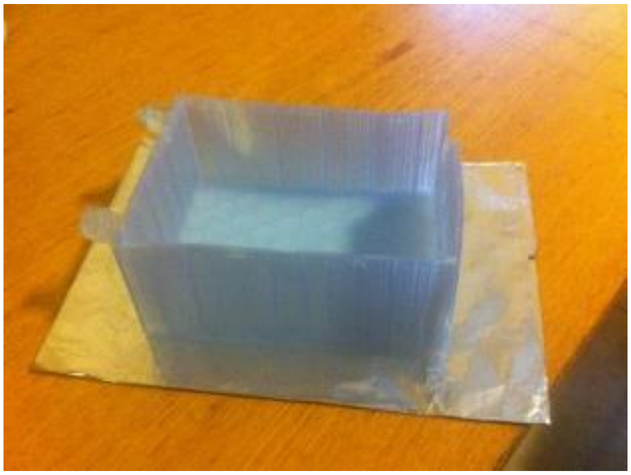
Results: No real effect. Warm and slightly malleable with force

At 190°F



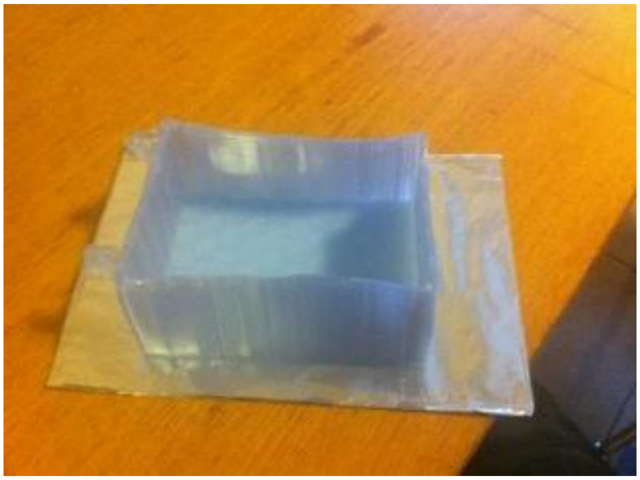
Results: PLA begins to become slightly malleable, corners are still rigid but the center walls begin to bow in with some force. Upon cooling it becomes stable again, with the walls still bowed.

At 210°F



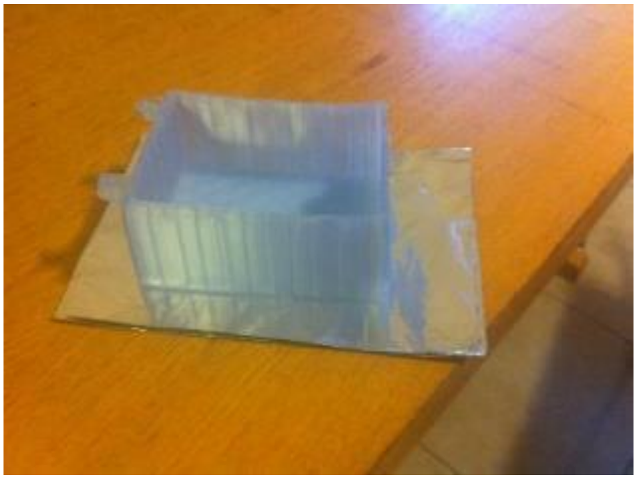
Results: PLA becomes malleable, outlying pieces and center walls can be bent up, however the corners are still rigid. (The wall that was bent slightly from the previous test, flattened back out when it was heated up.)

At 230°F



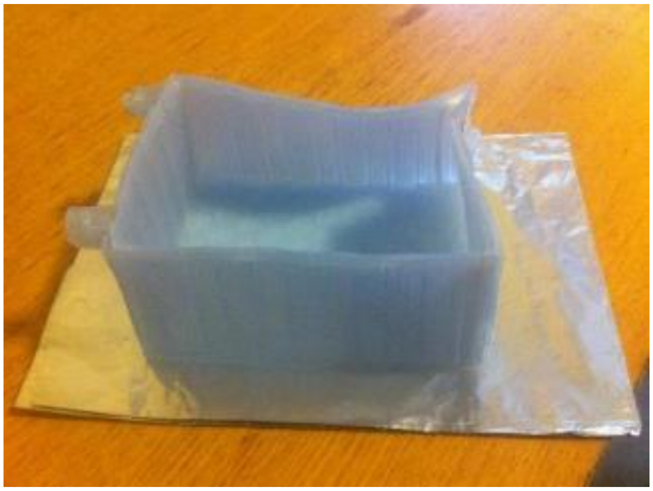
Results: Highly malleable center walls, corners are still fairly rigid, there is no melting or strand deformation.

At 250°F



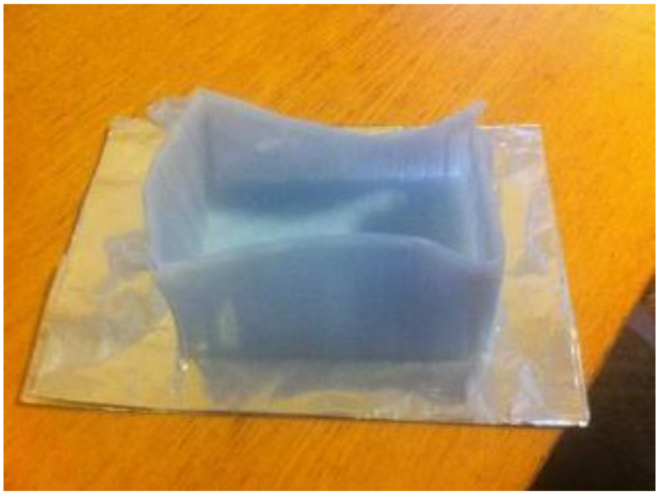
Results: Highly malleable center walls, corners became unstable, no melting or strand deformation.

At 270°F



Results: Highly malleable structural integrity seems to be diminishing, it feels as though I could ball it up, but there is not melting and no strand deformation.

At 290°F



Results: Similar to previous test is getting slightly less malleable. No melting or strand deformation.

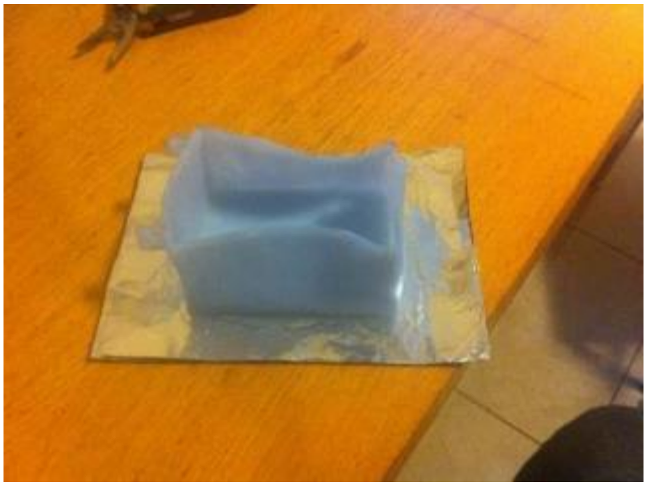
At 310°F ← too hot to handle at this point.

Results: Feels as though it is rubber, it seems like its hardening. Still no strand deformation or melting.

At 330°F

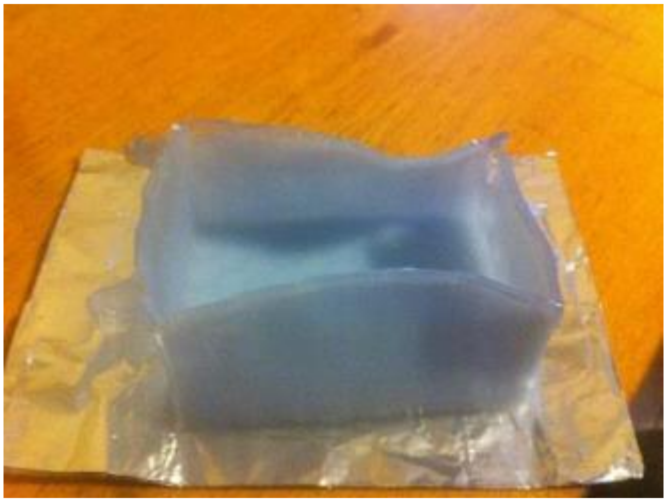
Results: The piece is becoming harder, and much less malleable. No strand deformation or melting. (I'm nervous it will soon catch fire and not melt slightly.)

At 350°F



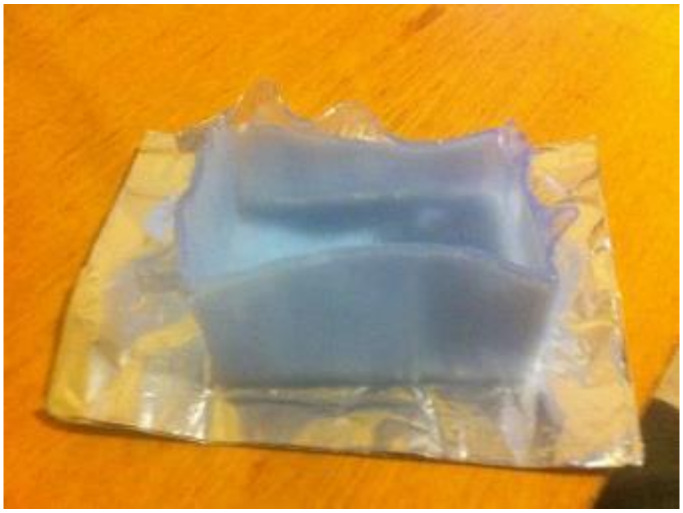
Results: Becoming rigid, very center of walls can be bent, but corners are sturdy and Very slight melting at broken corner. Strand deformation can occur on the melted area.

At 370°F



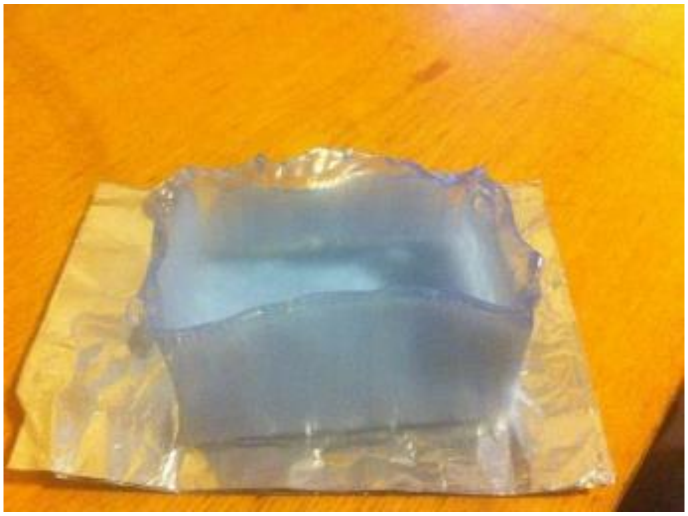
Results: Melting is occurring; the spots that have melted are now transparent. When touching the walls, the ribs are easy to feel. The melted parts are transparent after cooling. The strand deformation can occur on the melted area.

At 390°F



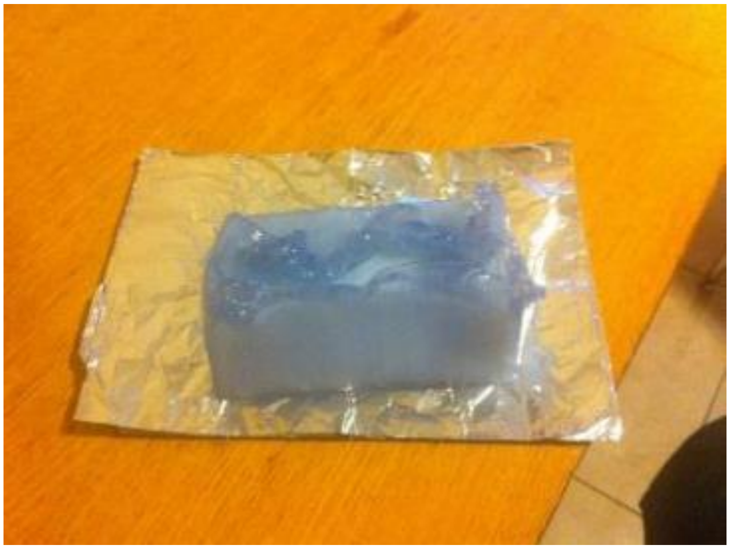
Results: Similar to previous test, strand deformation can occur if force is applied to the melted area. Transparency remains, there is still no charring.

At 410°F



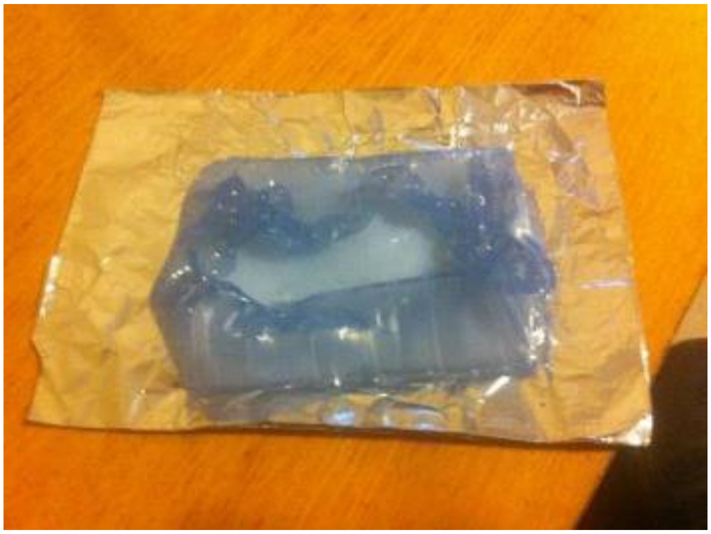
Results: more melting that previous test, strand deformation can occur more easily. Base has begun to melt in a few areas. It is becoming malleable again.

At 430°F



Results: Extreme melting, part of the box collapsed (the side closest to the heat source). Structural integrity is completely lost as the side melted to the base.

At 450°F

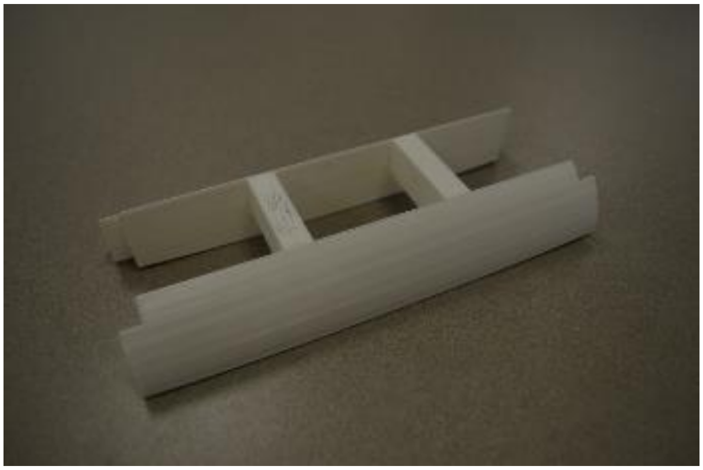


Results: Melted beyond point of recognition. No charring occurred, however this would not support any weight at all.

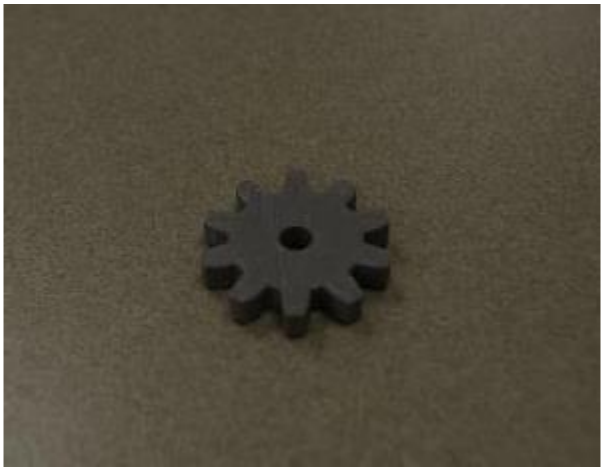
END TEST

Appendix C: Hexapod Model SolidWorks Parts

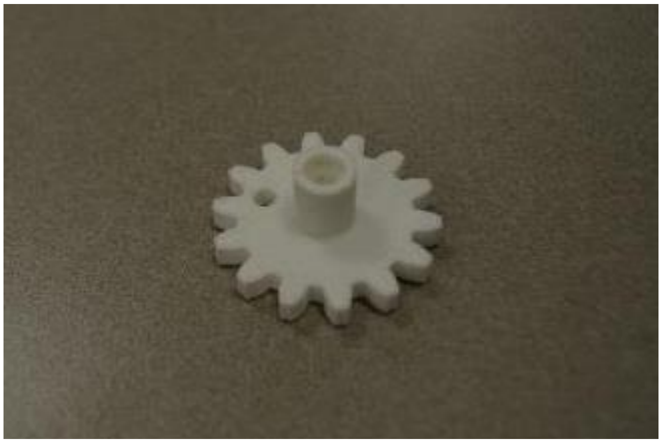
1) H-shape frame component



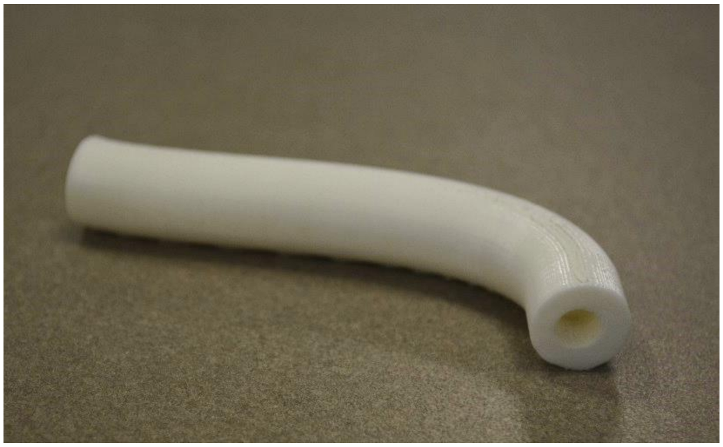
2) Idler gear



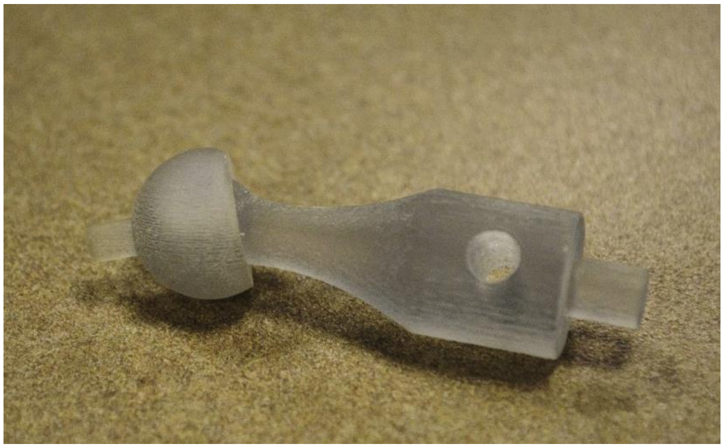
3) Driving gear



4) Lower leg



5) Ball-socket joint (transparent material)



Appendix D: Gear Motor Comparison

25D mm Gear Motors			
Load Torque	**RPMs**	**Stall torque**	**RPMs with Load**
52.5	9800	2	-247450
52.5	5600	1	-288400
52.5	2220	8	-12348.75
52.5	1280	5	-12160
52.5	1010	17	-2109.117647
52.5	580	11	-2188.181818
52.5	480	36	-220
52.5	275	24	-326.5625
52.5	285	60	35.625
52.5	165	40	-51.5625
52.5	210	80	72.1875
52.5	120	50	-6
52.5	130	130	77.5
52.5	75	85	28.67647059
52.5	100	160	67.1875
52.5	57	110	29.79545455
52.5	57	260	45.49038462
52.5	33	170	22.80882353
52.5	25	220	19.03409091
52.5	15	250	11.85
52.5	12	300	9.9
Micro Gear Motors			
Load Torque	**RPMs**	**Stall torque**	**RPMs with Load**
52.5	6000	2	-151500
52.5	3000	4	-36375
52.5	1000	9	-4833.333333
52.5	625	15	-1562.5
52.5	400	22	-554.5454545
52.5	320	30	-240
52.5	200	40	-62.5
52.5	140	50	-7
52.5	120	60	15
52.5	100	70	25
52.5	32	125	18.56

52.5	2200	3	-36300
52.5	730	8	-4060.625
52.5	420	13	-1276.153846
52.5	290	17	-605.5882353
52.5	220	19	-387.8947368
52.5	150	24	-178.125
52.5	2500	1	-128750
52.5	1300	2	-32825
52.5	440	4	-5335
52.5	250	7	-1625
52.5	170	9	-821.6666667
52.5	120	12	-405
52.5	85	17	-177.5
52.5	60	27	-56.66666667
52.5	50	32	-32.03125
52.5	45	40	-14.0625
52.5	14	70	3.5